Nutzerdefinierte Restriktion delegierter Privilegien im Grid-Computing

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

Doktor-Ingenieur
(Dr.-Ing.)

genehmigte

Dissertation

von
Dipl.-Ing. Stefan Piger
geboren am 06.06.1975 in München

2008

1. Referent	Prof. Dr.-Ing. Christian Grimm
2. Referent	Prof. Dr. techn. Wolfgang Nejdl
Tag der Promotion	15.12.2008

Vorwort

Die vorliegende Arbeit entstand während der Vorbereitung des BMBF-Verbundprojekts *D-Grid* und nachfolgend parallel zur Teilnahme daran im Rahmen meiner Tätigkeit am Regionalen Rechenzentrum für Niedersachsen der Leibniz Universität Hannover.

Mein ganz besonderer Dank gilt Herrn Prof. Grimm für das mir entgegengebrachte Vertrauen und die Gelegenheit, diese Arbeit sowie weitere Aufgaben in seiner Gruppe selbständig durchführen zu dürfen.

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

Meinen Kollegen Ralf Gröper und Christopher Kunz danke ich herzlich für die stets hilfreichen und anregenden Diskussionen zum Themenkomplex dieser Arbeit. Zudem danke ich ihnen und allen anderen aktuellen und ehemaligen Kollegen am Lehrgebiet Rechnernetze für die gute Zusammenarbeit sowie die persönliche und vertrauensvolle Arbeitsathmosphäre.

Astrid Tessmer und Gabriel Gaus danke ich für ihre Anmerkungen zu Orthographie und Grammatik respektive zur Gestaltung der Arbeit, wodurch deren Lesbarkeit deutlich erhöht wurde.

Mein letzter und größter Dank gebührt meiner Lebensgefährtin Edith Skach für die Unterstützung und den Rückhalt in Phasen der Anspannung sowie meinen Eltern, die mich in dem Beschluss bestärkt haben, die Promotion anzustreben.

Hannover, den 02.01.2009

Stefan Piger

Kurzfassung

In der vorliegenden Arbeit werden die in aktuellen Grid-Infrastrukturen im Einsatz befindlichen Verfahren zur Delegation von Nutzerprivilegien untersucht. Aus den Erkenntnissen dieser Analyse wird ein Konzept zur nutzerdefinierten Beschränkung von Delegationen abgeleitet, das in einer prototypischen Implementierung am Beispiel einer aktuellen Grid-Middleware umgesetzt wird.

Mit dem Konzept des Grid-Computing wird das Ziel verfolgt, verteilte Verbünde von Rechen- und Speicherressourcen für die Nutzung im eScience-Kontext zu etablieren sowie den Zugriff durch Nutzer auf diese zu vereinheitlichen. Weitergehende Vorstellungen sehen zukünftig eine direkte Kopplung von wissenschaftlichen Instrumenten an die Grid-Infrastruktur vor. Damit soll eine Nutzung der Geräte von entfernten Standorten aus erleichtert und der Zugriff auf dort generierte Daten für die Weiterverarbeitung über standardisierte Schnittstellen ermöglicht werden.

Den Kernbestandteil eines Grid-Verbundes bildet die Grid-Middleware, die eine Abstraktionsschicht zwischen Infrastruktur und Ressource bildet. Neben der Bereitstellung von einheitlichen Schnittstellen liegen ihre Aufgaben in der Überwachung von und der Information über Ressourcen. Auch wird üblicherweise eine Lastverteilung zwischen gleichartigen Ressourcen realisiert sowie die Nutzung von Betriebsmitteln erfasst und abgerechnet. Da Grid-Infrastrukturen aufgrund ihres hohen Grades an Verteiltheit und der Verfügbarkeit bedeutender Ressourcen attraktive Angriffsziele darstellen, werden zur Gewährleistung eines stabilen und verlässlichen Betriebs fortschrittliche Sicherheitsmechanismen implementiert. Zu diesen gehören Verfahren zur obligatorischen Verschlüsselung von Kommunikationsbeziehungen und zur beidseitigen Authentifizierung von Nutzern und Ressourcen mittels digitaler Zertifikate.

Die Inanspruchnahme indirekt genutzter Ressourcen erfordert den Einsatz von Delegationsmechanismen. In aktuellen Grid-Middlewares basieren diese auf Proxy-Zertifikaten und ermöglichen es Diensten, im Namen und mit den Privilegien des delegierenden Nutzers zu handeln. Zum Schutz der Nutzer werden Delegationen mit einer begrenzten Gültigkeitsdauer ausgestellt. Werden diese jedoch kompromittiert, ist der Angreifer in der Lage, den kompletten Umfang der Privilegien des Ausstellers auszuüben. Eine Beschränkung der delegierten Privilegien bei Ausstellung der Delegation ist in aktuellen Grid-Middlewares nicht möglich.

Als direkte Auswirkung unterliegen Disziplinen mit hohen Sicherheitsanforderungen Beschränkungen bei der Nutzung institutionsübergreifender Grid-Infrastrukturen. Dies betrifft insbesondere Anwendergruppen aus den medizinischen oder biologi-

schen Disziplinen, die mit personenbezogenen Datensätzen arbeiten und somit hohen datenschutzrechtlichen Anforderungen unterliegen. Ein Missbrauch von Privilegien kann jedoch auch anderweitig gravierende Auswirkungen haben. Wird eine kompromittierte Delegation dazu missbraucht, große Mengen an Betriebsmitteln zu verbrauchen, kann dem Anwender die Nutzung in Rechnung gestellt oder zumindest von dem ihm zur Verfügung stehenden Kontingent abgezogen werden.

Im ersten Teil dieser Arbeit wird eine Analyse der Autorisierungsmechanismen einer aktuellen Grid-Middleware durchgeführt. Die bestehenden Mechanismen zur Delegation von Nutzerprivilegien werden in Hinblick auf Funktionalität und Sicherheit untersucht und Schwächen identifiziert. Nachfolgend werden allgemeingültige Kriterien für Ansätze zur beschränkten Delegation aufgestellt und begründet sowie bestehende Ansätze analysiert und auf Basis dieser Kriterien bewertet.

Aus den Ergebnissen der Analyse werden Anforderungen an einen umfassenden Ansatz zur beschränkten Delegation von Privilegien abgeleitet und formuliert. Ziel des Ansatzes ist die Minimierung des Schadenspotentials im Falle erfolgreicher Kompromittierungen von Delegationen, ohne jedoch die Kompatibilität zu den in aktuellen Grid-Verbünden genutzten Autorisierungsmechanismen aufzugeben oder deren Komplexität deutlich zu erhöhen. Die Methodik des Ansatzes wird hinsichtlich der agierenden Entitäten, der Dimension der Beschränkung sowie der unterstützten Objekte dargelegt und begründet. Aus der Methodik wird ein Format für Richtlinien abgeleitet, das Nutzer in die Lage versetzt, Delegationen mit dem für eine Aktion im Grid benötigten Umfang ihrer Privilegien zu spezifizieren. Der Ansatz sieht die Einbettung der Richtlinien in die zur Delegation verwendeten Proxy-Zertifikate sowie ihren Transport im Push-Verfahren zu den zu nutzenden Grid-Ressourcen vor. Auf diesen erfolgt die Durchsetzung der nutzerdefinierten Richtlinien durch spezialisierte Komponenten der Zugriffskontrolle. Im Rahmen des Ansatzes werden zudem Verfahren diskutiert, die den Nutzer effizient bei der Erstellung der für die beschränkte Delegation benötigten Richtlinien unterstützen.

Im zweiten Teil der Arbeit wird eine prototypische Implementierung des erarbeiteten Ansatzes am Beispiel der gLite-Middleware vorgestellt. Diese unterstützt exemplarisch die Beschränkung von Privilegien für den Zugriff auf Speicherressourcen sowie für die Durchführung von Rechenaufträgen. Ein Ziel der Implementierung ist die einfache Erweiterbarkeit zur Unterstützung zukünftiger Dienste, die durch die Verwendung offener Standards wie XACML und X.509-Zertifikaten gewährleistet wird. Zudem wird eine wirksame Unterstützung der Nutzer bei der Erstellung der Richtlinien durch eine Kombination aus Automatisierung und ergonomischen Nutzerschnittstellen realisiert. Die Implementierung wird abschließend in Bezug auf ihre Funktionalität und Sicherheit sowie die erreichte Performance der Infrastrukturkomponenten hin bewertet.

Schlagwörter: Beschränkte Delegation, Grid-Computing, Proxy-Zertifikate

Abstract

This thesis analyses current Grid infrastructures regarding the implemented mechanisms for the delegation of user rights. The results of this analysis are subsequently employed to design an approach for the user-based restriction of delegated rights. Finally, a prototypical implementation of the approach based on a current Grid middleware package is presented.

The Grid computing approach pursues the vision of inter-institutional computing and storage infrastructures using standardized access mechanisms. In the future, Grid infrastructures are envisioned to be enhanced by interconnecting existing computing environments with scientific instruments to simplify access to experimental and observational data for postprocessing.

The core of Grid infrastructures is Grid middleware, which establishes an abstraction layer between the Grid and individual resources. Grid middlewares —in addition to providing standard interfaces— implement mechanisms to monitor participating systems and provide information about them. Furthermore, load-balancing between resources as well as accounting and billing functionalities are commonly provided. Due to their distributed nature and the availability of substantial resources, Grid infrastructures are attractive attack targets. To guarantee stable and reliable operation, advanced security mechanisms are implemented to establish mandatory encryption of communication connections and mutual authentication of users and services.

The utilization of indirectly accessed resources by users requires the employment of delegation mechanisms. Current Grid infrastructures base their delegation functionality on proxy certificates which enable services to act in the users' name and with the complete scope of their rights. To protect users, the lifetime of proxy certificates is limited. Should they be compromised however, a succesful attacker may act with the complete scope of the user's right. Current Grid infrastructures offer no mechanisms to restrict the delegated rights.

As a direct consequence of these shortcomings, disciplines with high security requirements are subject to tight restrictions in using inter-institutional Grid infrastructures. This in particular affects communities in the biological and medical disciplines which are subjected to strict data privacy regulations when working with personal data. Besides this, abuse of rights can cause negative repercussions for users when significant amounts of resources are consumed as resource usage is typically accounted and the user may be invoiced for it.

The first part of this thesis conducts an analysis of the authorization and access control mechanisms in current Grid middleware. The existing mechanisms for the delegation of users' rights are investigated with focus on functionality and security, weaknesses are identified. Subsequently, universal criteria for approaches to restricted delegation are established and existing approaches are analyzed and evaluated on this basis.

The results of the analysis lead to requirements for a comprehensive approach to restricted delegation of rights. The approach is formulated with the objective of minimizing the potential damage caused by compromised delegations without breaking compatibility to existing authorization mechanisms or increasing the complexity of these tasks beyond manageability. It is subsequently motivated with regard to the acting entities, the dimensions of restriction as well as the targets supported. From this, a format for policies is derived that enables users to specify delegations restricting access to the privileges needed in the context of the intended action on the Grid. These user-based policies are integrated in certificate extensions of proxy certificates employed for delegating privileges to Grid resources. Thus, the user-based policies are pushed to the resources accessed where corresponding components for access control enforce these policies. The functionality of such components is presented in this thesis as well as suitable approaches for the efficient and user-friendly support for the users in specifying the required policies.

The second part of this thesis presents a prototypical implementation of the developed approach based on the gLite middleware. To demonstrate the feasibility of the presented approach, two services are enhanced to support access control enforcing user-based policies. The first restricts access to the objects present on storage resources, the second restricts the execution of compute jobs and thus access to computing resources. A focus of the implementation is to ensure extensibility for future services through the use of standards such as XACML and ITU-T X.509 certificates. Furthermore, effective support for users is ensured by utilizing complementary approaches for the specification and generation of policies. Finally, the implementation is evaluated with regard to functionality and security as well as the achieved performance of the infrastructural components.

Key words: restricted delegation, Grid computing, proxy certificates

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Abkürzungen

AA	Attribute Authority
AAA	Authentication, Authorization, Accounting
AAI	Authentifizierungs- und Autorisierungsinfrastruktur
AC	Attribute Certificate. Ein durch den VOMS verwendetes und in RFC3281[FH02] standardisiertes Format zur Übermittlung von beglaubigten Attributen zur Autorisierung.
ACL	Access Control List
AES	Advanced Encryption Standard
AJO	Abstract Job Object
ARPA	Advanced Research Projects Agency
ARPANET	Advanced Research Projects Agency Network
ATM	Asynchronous Transfer Mode
AUP	Acceptable Use Policy
BMBF	Bundesministerium für Bildung und Forschung
CA	Certification Authority
CAS	Community Authorization Service
CDF	Cumulative Distribution Function
CE	Computing Element
CERN	Conseil Européen pour la Recherche Nucléaire
CLI	Command Line Interface
CRL	Certificate Revocation List
D-Grid	D-Grid-Initiative. Ein durch das BMBF gefördertes Grid-Projekt.
DAC	Discretionary Access Control

DEC	Digital Equipment Corporation
DEISA	Distributed European Infrastructure for Supercomputing Applications
DFN-Verein	Verein zur Förderung eines Deutschen Forschungsnetzes e. V.
DGAS	Distributed Grid Accounting System, ehemals DataGrid Accounting System
DGI	D-Grid-Integrationsprojekt
DN	Distinguished Name
DNS	Domain Name System
DoS	Denial of Service
DSSA	Distributed System Security Architecture
EDG	European DataGrid Project
EEC	End-Entity Certificate. Das X.509-Zertifikat eines Nutzers, eines Dienstes oder eines Rechnersystems.
EEZ	End-Entity-Zertifikat, siehe auch EEC
EGEE	Enabling Grids for E-sciencE, ehemals Enabling Grids for E-science in Europe
EUGridPMA	European Grid Policy Management Authority
FAS	File Authorization Service
FQAN	Fully Qualified Attribute Name
FTP	File Transfer Protocol
FTS	File Transfer Service. Ein Dienst der gLite-Middleware zur Durchführung von verlässlichen Third-Party Transfers von Daten zwischen Storage Elements.
FZK	Forschungszentrum Karlsruhe
GFAL	Grid File Access Library
GIIS	Grid Index Information Service
GRAM	Grid Resource Allocation Manager
GridFTP	Grid File Transfer Protocol

GRIS	Grid Ressource Information Service
GSI	Grid Security Infrastructure
GSS-API	Generic Security Services Application Programming Interface
GUID	Globally Unique Identifier
HLRN	Norddeutscher Verbund für Hoch- und Höchstleistungsrechnen
HTTP	Hypertext Transfer Protocol
I-WAY	Information Wide Area Year
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
ITU-T	International Telecommunication Union
JDL	Job Description Language
LAN	Local Area Network
LB	Logging & Bookkeeping
LCAS	Local Centre Authorization Service
LCG	Large Hadron Collider Computing Grid
LCMAPS	Local Credential MAPping Service
LDAP	Lightweight Directory Access Protocol
LFC	LCG File Catalog
LFN	Logical File Name
LHC	Large Hadron Collider
LRMS	Local Resource Management System
LSF	Load Sharing Facility
MAC	Mandatory Access Control
MDS	Monitoring and Discovery Service
MPI	Message Passing Interface
NIST	National Institute of Standards and Technology

OASIS	Organisation for the Advancement of Structured Information Standards
OCSP	Online Certificate Status Protocol
OID	Object-Identifier
PAP	Policy Administration Point
PBS	Portable Batch System
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PIP	Policy Information Point
PKI	Public-Key-Infrastructure
PRIMA	Privilege Management and Authorization in Grid Computing Environments
R-GMA	Relational Grid Monitoring Architecture
RA	Registration Authority
RBAC	Role-Based Access Control
RFC	Request For Comment, Standard der IETF
RRZN	Regionales Rechenzentrum für Niedersachsen der Leibniz Universität Hannover
RSA	Asymmetrisches Kryptographieverfahren nach Ronald L. Rivest, Adi Shamir und Leonard Adleman
RSL	Resource Specification Language
SAML	Security Assertion Markup Language
SD	Service Discovery
SE	Storage Element, siehe auch SRM
SGE	Sun Grid Engine
SLA	Service Level Agreement
SMTP	Simple Mail Transfer Protocol
SoA	Source of Authority

SPoF	Single Point of Failure
SRM	Storage Resource Manager
SSH	Secure Shell
SSO	Single Sign-On
SURL	Storage Uniform Resource Locator
TCP	Transport Control Protocol
TSC	Time Stamp Counter
TURL	Transport Uniform Resource Locator
UI	User Interface
URL	Uniform Resource Locator
UTC	Universal Time Coordinated
VO	Virtuelle Organisation
VOMS	Virtual Organisation Membership Service
WAN	Wide Area Network
WMS	Workload Management System
WN	Worker Node
WWW	World Wide Web
XACML	eXtensible Access Control Markup Language

1 Einleitung

1.1 Einführung und Motivation

Die Verfügbarkeit einer robusten und leistungsfähigen Netzinfrastruktur im wissenschaftlichen und kommerziellen Umfeld gestattet seit den 70er Jahren des 20. Jahrhunderts den Zugriff auf entfernte Ressourcen der Informationstechnik. Ermöglicht wurde diese Entwicklung durch das *Internet Protocol* (IP) [Pos81a] und das darauf aufbauende Internet. Durch diese Technologie konnte erstmals über ein global verfügbares und vereinheitlichtes Protokoll der Vermittlungsschicht auf Kommunikationsdienste zugegriffen werden, die sich außerhalb des lokalen Netzwerkes (LAN) der eigenen Institution befanden.

Mit der zunehmenden Verbreitung von Netztechnologien und -protokollen verändert sich auch die Nutzung von Großrechenanlagen im wissenschaftlichen Umfeld. Während diese in den Anfängen nur auf lokaler Ebene genutzt wurden und die Bedienung ausschließlich durch speziell eingewiesene Operateure erfolgte, konnte der Zugriff auf Großrechner später von entfernten Rechnern aus über leistungsfähige Netze realisiert werden. Diese Entwicklungen kulminierten in der Zusammenschaltung von Großrechnern zu virtuellen Systemen über dedizierte bzw. speziell für diese Anwendungen optimierte Weitverkehrsnetze (WAN), wie dies im *Norddeutschen Verbund für Hoch- und Höchstleistungsrechnen* (HLRN) [HLR06] oder auch im europäischen DEISA-Projekt [DEI08] praktiziert wird. Diese gekoppelten Rechner bestehen zumeist aus weitgehend homogener Hardware mit identischem Betriebssystem und erscheinen dem Benutzer wie ein monolithisches System. Vorteile ergeben sich für diese virtuellen Hochleistungsrechner insbesondere durch die Fähigkeit, lokale Lastspitzen durch ein Umschichten von Rechenaufträgen, so genannter Jobs, auf entfernte Komponenten des Systems abzumildern. Nachteilig ist die mangelnde Flexibilität durch die enge administrative Kopplung. Eine Integration andersartiger Hardware, die unter Umständen unterschiedliche Betriebssysteme verwendet, ist zumeist unmöglich oder mit erheblichen Schwierigkeiten verbunden.

Das Grid-Computing zielt darauf ab, die Nachteile bisheriger Großrechnerverbünde zu beheben. Insbesondere sollen bestehende Systeme – in aktuellen Infrastrukturen sind dies in der Regel Rechencluster sowie Speicherressourcen unterschiedlicher Architekturen – effizient gekoppelt werden können. Dabei soll die Ressourcenauswahl für den Nutzer transparent erfolgen. Ein weiteres Ziel ist es, Nutzern einheitliche Mechanismen für den Zugang zu den am Grid beteiligten Ressourcen zur Verfügung zu stellen, wodurch die übliche Lernphase vor der produktiven Nutzung von Großrechnern entfallen soll.

Die für das Erreichen der genannten Ziele benötigten Funktionalitäten stellt der Kernbestandteil einer Grid-Infrastruktur bereit, die Grid-Middleware. Diese bildet eine Abstraktionsschicht zwischen Grid-Infrastruktur und Ressource, auf denen sie standardisierte Schnittstellen etabliert. Neben der Bereitstellung eines einheitlichen Zugangsverfahrens für Nutzer liegen ihre Aufgaben in den Bereichen der Überwachung von und Information über Ressourcen. Darüber hinaus wird üblicherweise eine Lastverteilung zwischen gleichartigen Ressourcen realisiert sowie die Nutzung von Betriebsmitteln erfasst und abgerechnet.

Grid-Infrastrukturen stellen aufgrund ihres hohen Grades an Verteiltheit über Institutions- und Ländergrenzen hinweg sowie der Verfügbarkeit bedeutender Ressourcen attraktive Angriffsziele dar. So sind die am *Large Hadron Collider Computing Grid* (LCG) [LHC06] teilnehmenden Institutionen über mehrere Kontinente verteilt und verfügen in Summe über einige 10000 Prozessoren sowie über mehrere Petabyte an Speicherkapazität. Um einen stabilen und verlässlichen Betrieb der Grid-Infrastrukturen zu gewährleisten sowie den Missbrauch von Ressourcen zu erschweren, implementieren Grid-Middlewares fortschrittliche Sicherheitsmechanismen. Dazu gehören Verfahren sowohl zur obligatorischen Verschlüsselung von Kommunikationsbeziehungen als auch zur gegenseitigen Authentifizierung von Nutzern und Ressourcen mittels digitaler Zertifikate.

Um eine Inanspruchnahme indirekt genutzter Ressourcen zu ermöglichen, implementieren die auf dem *Globus Toolkit* aufbauenden Grid-Middlewares Delegationsmechanismen. Diese kommen zum Einsatz, wenn Nutzer im Rahmen von nichtinteraktiven Aktivitäten auf weitere Grid-Ressourcen zugreifen wollen. Die verwendeten Delegationsverfahren basieren auf Proxy-Zertifikaten und ermöglichen es Grid-Diensten, für einen begrenzten Zeitraum im Namen der Nutzer zu handeln. Während dieses Zeitraums kann der Empfänger einer Delegation den vollständigen Umfang der Privilegien des Ausstellers ausüben. Da eine Beschränkung der delegierten Privilegien in aktuellen Grid-Middlewares nicht implementiert ist, gilt dies bei einer Kompromittierung der Delegation auch für den erfolgreichen Angreifer, dem somit ebenfalls alle Privilegien des Ausstellers zur Verfügung stehen.

Als direkte Auswirkung dieses Mangels unterliegen Disziplinen mit hohen Sicherheitsanforderungen Beschränkungen bei der Nutzung institutionsübergreifender Grid-Infrastrukturen. Dies betrifft insbesondere Anwendergruppen mit hohen datenschutzrechtlichen Anforderungen, z. B. aus den medizinischen oder biologischen Disziplinen, die mit personenbezogenen Datensätzen arbeiten. Ein Missbrauch von Privilegien kann jedoch auch anderweitig gravierende Auswirkungen haben. Wird eine kompromittierte Delegation dazu missbraucht, große Mengen an Betriebsmitteln zu verbrauchen, kann dem Anwender diese Nutzung in Rechnung gestellt oder zumindest von dem ihm zur Verfügung stehenden Kontingent abgezogen werden.

1.2 Zielsetzung

Die Zielsetzung der Arbeit leitet sich aus den in der Einführung skizzierten Schwächen der implementierten Delegationsmechanismen ab. Im Rahmen dieser Arbeit soll ein Ansatz für die wirksame Beschränkung der delegierten Privilegien entwickelt werden. Dieser Ansatz soll gewährleisten, dass nur die für die intendierten Aktivitäten benötigten Privilegien delegiert werden, um das Schadenspotential im Falle einer Kompromittierung zu minimieren.

In einem ersten Schritt ist dafür eine systematische Analyse der etablierten Authentifizierungs- und Autorisierunginfrastruktur durchzuführen. Darüber hinaus sind die zum Schutz der Delegationen implementierten Sicherheitsmechanismen der Grid-Ressourcen zu untersuchen, um das Risiko einer Kompromittierung realistisch einschätzen zu können. Aus den gewonnenen Erkenntnissen sollen allgemeingültige Kriterien für einen umfassenden und implementierbaren Ansatz abgeleitet werden, anhand derer nachfolgend bestehende Ansätze bewertet werden.

Aus den Ergebnissen der Analyse wird der Ansatz für die beschränkte Delegation entwickelt. Neben dem funktionalen Ziel einer effektiven Beschränkung der delegierten Privilegien sind weitere Anforderungen zu erfüllen. So ist zu gewährleisten, dass die in der Grid-Infrastruktur eingesetzten Dienste durch diesen Ansatz unterstützt werden können. Auch ist darauf zu achten, dass die durch den Ansatz erzeugte zusätzliche Komplexität für alle Beteiligten beherrschbar bleibt. Dies gilt sowohl für die die Delegation erstellenden Nutzer als auch für die Attributautoritäten und die Betreiber der Grid-Ressourcen. Diese Randbedingung resultiert aus der Erfahrung, dass Sicherheitsmechanismen nur Anwendung finden, wenn der Aufwand im Verhältnis zum erkennbaren Nutzen gering ist.

Im zweiten Schritt soll der entwickelte Ansatz beispielhaft für eine Grid-Middleware umgesetzt werden, um seine Praxistauglichkeit zu verifizieren. Für die Implementierung soll die gLite-Middleware verwendet werden, die sich durch eine Vielfalt an bereitgestellten Diensten auszeichnet und somit eine vollständige Umsetzung des entwickelten Ansatzes ermöglicht. Zudem wird eine Implementierung durch die vertieften Kenntnisse der gLite-Middleware unterstützt, die das Lehrgebiet Rechnernetze der Leibniz Universität Hannover durch seine Beteiligung als assoziierter Partner an den Arbeiten des Projekts *Enabling Grids for E-sciencE* (EGEE) [EGE07] im Bereich der Aktivität *JRA1-testing* erwerben konnte. Neben der Validierung des Ansatzes soll die Implementierung die wichtigsten Dienstarten der gLite-Middleware unterstützen sowie eine spätere Erweiterung für zusätzliche Dienste ermöglichen. Weiterhin ist die Kompatibilität zu den Mechanismen der Authentifizierung und Autorisierung der gLite-Middleware zu wahren, um die Einführung in bestehende Infrastrukturen zu vereinfachen.

1.3 Aufbau der Arbeit

In Kapitel 2 werden die für die weitere Arbeit benötigten Grundlagen dargelegt. Nach einer Einführung in das Grid-Computing und aktuelle Grid-Middlewares stehen Verfahren der Authentifizierung und Autorisierung im Fokus. Die Ausführungen beschränken sich dabei auf die im Grid-Umfeld verwendeten Verfahren.

Kapitel 3 ist der Analyse von Autorisierungsvorgängen in Grid-Umgebungen gewidmet. Betrachtet werden neben den Abläufen zur Laufzeit von Rechenaufträgen die in der gLite-Middleware implementierten Delegationsmechanismen. Im dritten Abschnitt des Kapitels werden Kriterien aufgestellt, anhand derer diese Mechanismen mit Delegationsverfahren verglichen werden, die außerhalb von gLite-Umgebungen entwickelt werden bzw. bereits im Einsatz sind.

Kapitel 4 stellt den aus den gewonnenen Einsichten entwickelten Ansatz vor. Dabei wird auf das Konzept der Selbstbeschränkung der Nutzer eingegangen und die dafür verwendeten nutzerdefinierten Richtlinien werden vorgestellt. Weiterhin werden die für eine wirksame Zugriffskontrolle benötigten Implementierungsvarianten diskutiert sowie Ansätze für eine effektive Unterstützung der Nutzer bei der Erstellung der Richtlinien vorgestellt.

Kapitel 5 behandelt die prototypische Implementierung des in Kapitel 4 diskutierten Ansatzes. Nach einer Übersicht über die enthaltenen Komponenten werden die in XACML codierten Richtlinien und ihre Integration in Proxy-Zertifikate vorgestellt. Es folgen Ausführungen, die die Generierung der Richtlinien sowie die zur Unterstützung der Nutzer entwickelten Komponenten beschreiben. In den letzten drei Abschnitten des Kapitels werden die implementierten Zugriffskontrollmechanismen anhand der nutzerdefinierten Richtlinien vorgestellt sowie Erweiterungen des MyProxy-Dienstes erläutert.

Kapitel 6 nimmt eine Bewertung der durchgeführten Implementierung vor. Dabei wird die Funktionalität anhand der in Kapitel 4 aufgestellten Kriterien bewertet, zudem werden Aspekte der Sicherheit der implementierten Mechanismen sowie der durch diese verursachte Overhead diskutiert.

Mit einer Zusammenfassung in Kapitel 7 schließt die Arbeit. Darin werden die Ergebnisse der Arbeit aufgeführt sowie ein Ausblick auf zukünftige Einsatzbereiche und Entwicklungsrichtungen für den erarbeiteten Ansatz gegeben.

In Anhang A findet sich ein Überblick der für die praktischen Arbeiten genutzten Grid-Umgebung sowie der enthaltenen Systeme und Dienste. Anhang B stellt den Ablauf der Erstellung eines exemplarischen Rechenauftrags unter Nutzung der webbasierten Nutzerschnittstelle dar.

2 Grundlagen

2.1 Das Grid-Computing

Der Begriff Grid-Computing wurde in den 1990er Jahre durch Ian Foster und Carl Kesselman geprägt [KF98]. Er wurde in Analogie zum Stromnetz gewählt, das im amerikanischen Sprachgebrauch als *Power-Grid* bezeichnet wird. Dieses stellt dem Nutzer nahezu permanent und ubiquitär einen Zugang zu Elektrizität zur Verfügung und ist ohne tiefes Verständnis seines Aufbaus nutzbar. Grid-Computing soll dem Nutzer in ähnlicher Weise Ressourcen der Informationstechnologie zugänglich machen, d. h. es soll die Nutzung der Ressourcen ohne genauere Kenntnisse über den Aufbau, die Funktionsweise oder den Standort der Ressource ermöglichen.

Grid-Computing zielt insbesondere darauf ab, verteilte Verbünde von Rechen- und Speicher-Ressourcen für die Nutzung im wissenschaftlichen Kontext aufzubauen [FK03] sowie den Zugriff auf diese zu vereinheitlichen. Weitergehende Vorstellungen sehen in Zukunft eine direkte Kopplung wissenschaftlicher Instrumente, wie z. B. Radioteleskope, an die Grid-Infrastruktur vor. Damit soll eine Nutzung der Geräte von entfernten Standorten erleichtert und der Zugriff auf dort produzierte Messdaten über eine einheitliche Schnittstelle ermöglicht werden. Generell wird angestrebt, das Grid zu einem Kernbestandteil der auf Informationstechnik gestützten Wissenschaften, die auch als *eScience* bezeichnet werden, weiterzuentwickeln.

2.1.1 Virtuelle Organisationen

Herkömmliches Supercomputing basiert – unabhängig davon, ob es sich um einen einzelnen Rechner oder um einen Verbund von Großrechnern handelt – auf einem statischen Ansatz. Eine installierte und konfigurierte Infrastruktur wird in der Regel nur noch geringfügig verändert.

Grid-Infrastrukturen sind hingegen per definitionem dynamisch, d. h. es ist vorgesehen, dass Organisationen dem Grid-Verbund beitreten und ihn ebenso auch wieder verlassen können, ohne die Infrastruktur grundlegend überarbeiten zu müssen. Diese Dynamik umfasst naturgemäß nicht nur die Infrastruktur des Grids, sondern in höherem Maße noch die Nutzerschaft, die sich aus den am Grid beteiligten Institutionen rekrutiert. Den Nutzern soll nicht – wie in herkömmlichen Großrechner-Umgebungen üblich – in langwierigen Antragsverfahren einzeln ein Zugang zu den Rechen- und Speicher-Ressourcen erteilt werden. Es wird angestrebt, ganzen Communities, die ein gemeinsames Interesse und Arbeitsziel definiert, den Zugang zum

Grid zu gestatten. Diese Communities wurden von Foster et al. in [FKT01] definiert und als *Virtuelle Organisationen* (VO) bezeichnet:

> „... First, we review the "Grid problem", which we define as flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources – what we refer to as virtual organizations."

Bei einer VO handelt es sich folglich um eine Gruppierung von Institutionen oder Einzelpersonen sowie eine Anzahl von Ressourcen oder zumindest das Recht, diese zu nutzen. Eine Grundeigenschaft von Virtuellen Organisationen ist somit die gemeinsame Nutzung zur Verfügung stehender Ressourcen. Ein weiteres Bindeglied zwischen den beteiligten Entitäten ist eine Vereinbarung, welche Regeln beim Zugriff auf gemeinsam genutzte Ressourcen zu beachten sind und was das gemeinsame Ziel ist. Foster et al. schreiben in [FKT01] dazu:

> „... 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."

Mit der Definition eines gemeinsamen Ziels ist zugleich die Endlichkeit der Virtuellen Organisation festgelegt. Eine Virtuelle Organisation löst sich mit Erreichen des gemeinsamen Ziels auf.

2.1.1.1 Beispiel einer Grid-Infrastruktur mit realen und Virtuellen Organisationen

In Abbildung 2.1 auf der nächsten Seite ist ein Zusammenschluss von drei Virtuellen Organisationen dargestellt, die sich aus Angehörigen und Ressourcen von drei realen Institutionen zusammensetzen. Am Beispiel dieser Abbildung sollen einige Grundprinzipien von Virtuellen Organisationen verdeutlicht werden. *VO1* besteht aus Ressourcen und Angehörigen von *Institution A* und *Institution C* sowie Ressourcen von *Institution B*. Eine Institution wie hier die *Institution B* kann folglich auch als reiner Ressourcen-Provider an einer Virtuellen Organisation beteiligt sein und muss keineswegs immer auch Mitarbeiter stellen. *VO2* setzt sich aus Mitarbeitern und Ressourcen von *Institution A* sowie Ressourcen von *Institution B* und Angehörigen von *Institution C* zusammen. An diesem Beispiel soll verdeutlicht werden, dass ebenso, wie nur Ressourcen von einer Institution in eine VO eingebracht werden können, auch lediglich Mitarbeiter den Beitrag einer Institution zu einer VO darstellen können. Weiterhin können Ressourcen, wie hier am Beispiel von *Institution B*, auch in mehrere Virtuelle Organisationen zur selben Zeit eingebracht werden. Das Beispiel von *VO3* verdeutlicht darüber hinaus, dass auch Angehörige einer Institution – wie hier in *Institution C* – Teilnehmer an mehr als einer Virtuellen Organisation zur selben Zeit sein können.

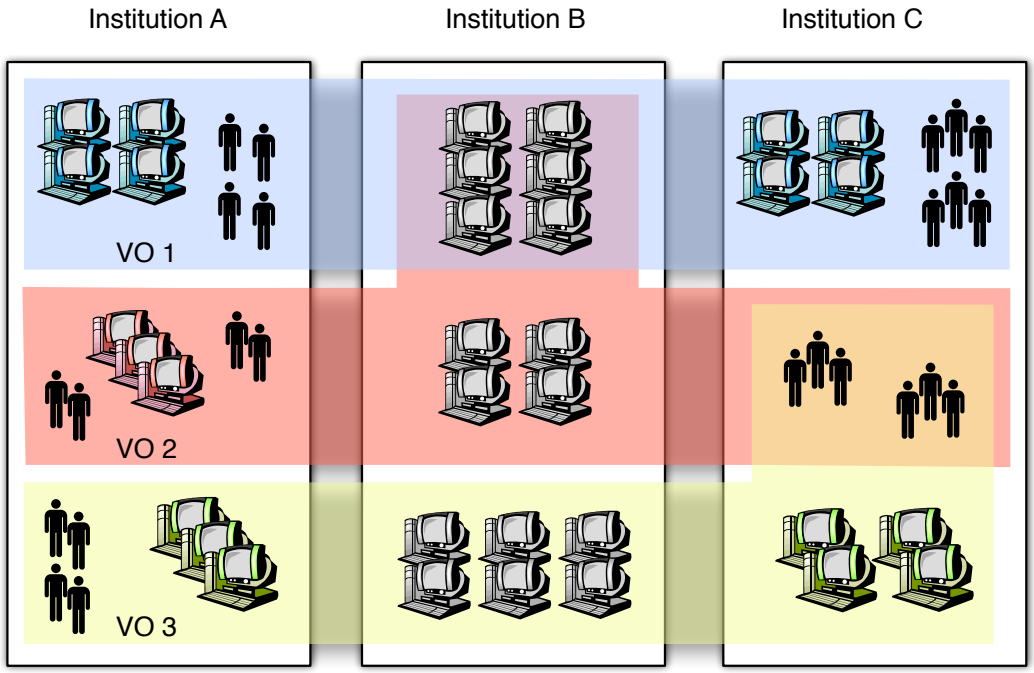


Abbildung 2.1: Virtuelle Organisationen

2.1.2 Das Globus Toolkit

Die Basis von vielen aktuell verwendeten Software-Komponenten in Grid-Infrastrukturen ist das amerikanische *Globus Toolkit 2* (GT2), dessen Architektur in Abbildung 2.2 auf der nächsten Seite abgebildet ist. Eine Weiterentwicklung von GT2 stellt die Version 4 des *Globus Toolkit* [Fos05, Fos07] dar, die im April 2005 veröffentlicht wurde. Die bedeutendste Neuerung der Version 4 ist die weitgehende Verwendung von *Web Services* zur Kommunikation zwischen Grid-Ressourcen. Aufgrund der fundamentalen Bedeutung von GT 2 wird hier lediglich auf die ältere Version eingegangen.

Die Entwicklung des *Globus Toolkit* [FK97, FK99] hat ihre Grundlagen im US-amerikanischen Projekt *Information Wide Area Year* (I-WAY) [DFP+96]. I-WAY hatte das Ziel, amerikanische Rechenzentren per WAN-Verbindungen auf ATM-Basis miteinander zu verknüpfen, um Wissenschaftlern den Zugang zu Großrechner- und Visualisierungsressourcen zu erleichtern.

Das *Globus Toolkit* versucht nicht, eine komplette Grid-Middleware zu implementieren. Es stellt lediglich eine Anzahl von fundamentalen Funktionalitäten zur Verfügung, die für eine Grid-Infrastruktur benötigt werden. Zu diesen zählen insbesondere folgende Mechanismen:

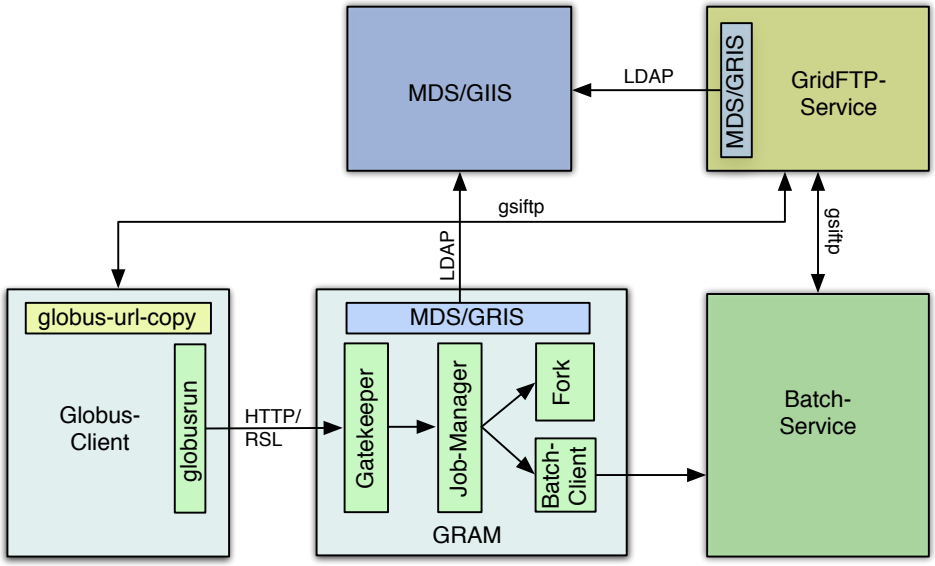


Abbildung 2.2: Architektur des Globus Toolkit 2

- Information über die verfügbaren Ressourcen und ihren Zustand mittels des *Monitoring and Discovery Service* (MDS).
- Management von Rechenaufträgen. Dessen Basis ist der *Grid Resource Allocation Manager* (GRAM).
- Dateitransfer zwischen Grid-Ressourcen mit hoher Datenrate durch Verwendung von GridFTP, einer Erweiterung des *File Transfer Protocol* (FTP) [PR85].
- Authentifizierung von Nutzern und Diensten. Diese Funktionalität wird auf Basis von X.509-Zertifikaten und Public-Key-Infrastrukturen (PKI) (vgl. Kapitel 2.2.2) durch die *Grid Security Infrastructure* (GSI) (vgl. Kapitel 2.2.3) bereitgestellt.

MDS Der MDS besteht aus einer Komponente auf jeder Ressource, dem *Grid Ressource Information Service* (GRIS) sowie einem zentralen Dienst, dem *Grid Index Information Service* (GIIS). Der GRIS ist für die Informationen über den Zustand der einzelnen Ressource zuständig. Die dafür benötigten Informationen werden durch spezielle *Information Provider*, die auf der Ressource arbeiten, zur Verfügung gestellt. Der GIIS fragt in regelmäßigen Abständen die GRIS-Komponenten ab und macht die gesammelten Informationen dem Nutzer zugänglich. Beide Monitoring-Dienste basieren auf dem *Lightweight Directory Access Protocol* (LDAP).

GRAM Der GRAM nimmt Rechenaufträge der Nutzer entgegen und leitet sie an die Grid-Ressourcen weiter. Die Aufträge werden in der *Resource Specification Language* (RSL) formuliert, die Kommunikation erfolgt über HTTPS. Der Dienst besteht aus zwei Server- und einer Client-Komponente:

- **Gatekeeper.** Diese Server-Komponente dient dazu, Aufträge vom Klienten anzunehmen und an den Job-Manager weiterzuleiten.
- **Job-Manager.** Hierbei handelt es sich um einen einfachen Job-Scheduler. Dieser ist in der Lage, Aufträge direkt auf der Maschine auszuführen, auf der der GRAM-Dienst installiert ist (Fork-Queue). Alternativ kann er die Aufträge an ein externes Batch-System (z. B. PBS) weiterleiten.
- **globusrun.** Das Programm *globusrun* wird verwendet, um Aufträge des Nutzers an einen GRAM-Dienst zu übergeben.

GridFTP GridFTP [ABB+03] ist eine Erweiterung des klassischen FTP um starke Authentifizierung mittels der von der GSI zur Verfügung gestellten Mechanismen. Weiterhin unterstützt GridFTP die Verteilung des Datenstroms auf mehrere TCP-Verbindungen zur Erhöhung der erreichbaren Datenrate. Zusätzlich können mittels GridFTP so genannte *Third-Party Transfers* durchgeführt werden. Bei diesen handelt es sich um Übertragungen von Dateien zwischen zwei GridFTP-Servern, die von einem Klienten aus gesteuert werden.

2.1.3 Europäische Grid-Projekte und die gLite-Middleware

Die Europäische Union (EU) fördert die Entwicklung von Grid-Middleware und den Betrieb von Grid-Infrastrukturen seit 2001. Aus den Projekten *European DataGrid* (EDG) und *Enabling Grids for E-sciencE* (EGEE) mit seinem Nachfolger EGEE-II gingen die Grid-Middleware *LHC Computing Grid 2* (LCG2) und *gLite* hervor. Die LCG2-Middleware wird nicht mehr weiterentwickelt, ihre Komponenten sind jedoch vollständig in der Version 3.0 der gLite-Middleware enthalten.

Im Gegensatz zu den Zielen der Entwickler des *Globus Toolkit* sollten LCG2 und die gLite-Middleware nicht nur die Basisfunktionalität für eine Grid-Infrastruktur bereitstellen. Die Projekte zielten darauf ab, eine produktive Grid-Infrastruktur für die europäische Wissenschaft aufzubauen. Diese Zielsetzungen wurden insbesondere durch die Planungen für den *Large Hadron Collider* (LHC) am europäischen Kernforschungszentrum CERN beeinflusst. Dessen Ausgabedaten sollen über die europäische Grid-Infrastruktur den beteiligten Instituten zugänglich gemacht und dort weiterverarbeitet werden. Die Hochenergiephysik ist damit der wichtigste Anwendungsbereich für die europäischen Grid-Projekte, jedoch haben diese explizit den Auftrag, weitere Disziplinen bei der Verwendung des Grids zu unterstützen.

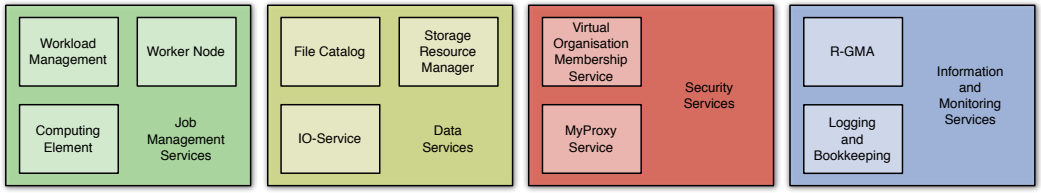


Abbildung 2.3: Dienstegruppen der gLite-Middleware

Die Zielsetzung, eine komplette Grid-Middleware zu entwickeln, schaffte die Notwendigkeit, gegenüber dem *Globus Toolkit* weitere Funktionsbereiche abzudecken. Die gLite-Middleware in der Version 3.0 wird aus vier Gruppen von Diensten (vgl. Abbildung 2.3) gebildet, die jeweils aus mehreren Komponenten bestehen. Diese vier Dienstegruppen werden in den folgenden Abschnitten vorgestellt. Eine detaillierte Analyse der für diese Arbeit relevanten Dienste wird in Kapitel 3 vorgenommen.

2.1.3.1 Job Management Services

Grid-Computing befasst sich mit der Bearbeitung von – in der Regel nicht-interaktiven – Rechenaufträgen, so genannten Batch-Jobs. Die Funktionalität, diese Art von Rechenaufträgen auf Grid-Ressourcen auszuführen, ist die Basis der Grid-Middleware. In *gLite* ist diese Funktionalität auf eine Gruppe von drei Komponenten verteilt. Diese besteht aus *Workload Management System* (WMS), *Computing Element* (CE) sowie dem *Worker Node* (WN).

Workload Management System Ein Grid besteht – wie schon in Abschnitt 2.1 dargelegt – aus einer Anzahl miteinander gekoppelter, räumlich verteilter Ressourcen. Vor der eigentlichen Inanspruchnahme der Dienste einer Grid-Infrastruktur steht für den Nutzer die Aufgabe, eine für ihn passende, freie Ressource auszuwählen. Das WMS soll den Nutzer von dieser Aufgabe befreien, indem es nach den Spezifikationen des Nutzers automatisch die passende Ressource für seinen Auftrag findet. Dafür erhält das WMS von den Grid-Ressourcen sowie aus den Informationsdiensten Daten über die zur Verfügung stehenden Ressourcen und ihren Auslastungszustand. In einer auf *gLite* aufbauenden Grid-Infrastruktur muss mindestens ein WMS vorhanden sein. Es können auch mehrere Komponenten dieses Typs parallel eingesetzt werden, um die Leistungsfähigkeit der Infrastruktur bei einer großen Anzahl simultaner Batch-Jobs zu verbessern oder die Fehlertoleranz zu erhöhen.

Computing Element Das CE stellt die Schnittstelle zwischen Grid-Infrastruktur und lokalen Ressourcen dar. Es nimmt Rechenaufträge des WMS entgegen und

leitet sie an die eigenen Ressourcen weiter. Diese als *Worker Nodes* bezeichneten Rechenknoten sind in der Regel durch ein *Local Resource Management System* (LRMS) an das CE gekoppelt. Als LRMS kommen Batch-Systeme wie das *Portable Batch System* (PBS) oder die *Sun Grid Engine* (SGE) zum Einsatz.

Worker Node Der WN ist für die Bearbeitung der Rechenaufträge des Nutzers zuständig. Er stellt im eigentlichen Sinne keine Komponente der Grid-Infrastruktur dar, da er lediglich über ein LRMS angebunden ist und keine Grid-Dienste zur Verfügung stellt. In Umgebungen, die auf *gLite* aufbauen, sind auf dem WN jedoch Softwarekomponenten installiert, mit denen die Datendienste des Grid genutzt werden können. So lassen sich aus einem Rechenauftrag des Nutzers heraus Daten von einem Grid-Service abrufen. Auch werden Ergebnisdateien des Rechenauftrags über Grid-Komponenten zum Nutzer übertragen. Durch diese Funktionalitäten nimmt der WN auch am Grid teil.

2.1.3.2 Data Services

Rechenaufträge der Nutzer von Grid-Umgebungen benötigen in der Regel Eingabedaten, zudem werden bei der Abarbeitung Ausgabedaten erzeugt. Um die Komponenten des Job-Management, insbesondere das WMS und das CE, nicht mit Datentransfer und -speicherung zu belasten, übernehmen spezielle Komponenten diese Aufgabe. Neben der reinen Speicherung von Nutzerdaten gehört dazu auch die Katalogisierung der Speicherorte und der Zugriffsrechte auf die Daten. Die genannten Aufgaben werden innerhalb der gLite-Middleware von den *Data Services* erfüllt. Diese bestehen aus den Komponenten *File Catalog*, *Storage Resource Manager* (SRM) sowie *IO-Service*.

File Catalog Der *File Catalog* hat die Aufgabe, Informationen über die Dateien des Nutzers zu verwalten. Diese Informationen umfassen den logischen Dateinamen (*Logical File Name*, LFN), ihren Speicherort auf den Grid-Ressourcen als *Storage URL* (SURL), evtl. vorhandene Repliken sowie Zugriffsrechte auf diese Datei. Diese Funktionalität soll den Nutzer davon entlasten, eigene Datenbanken für die Verwaltung seiner Dateien auf Grid-Ressourcen führen zu müssen. Der *File Catalog* präsentiert dem Nutzer seine Verzeichnisse in einer hierarchischen Baumstruktur, die an UNIX-Verzeichnisstrukturen angelehnt ist.

Storage Resource Manager Der SRM-Service [Sim07] ist sowohl eine Middleware-Komponente als auch eine Interface Definition [SSP+07]. Der SRM wird zur Vereinheitlichung des Zugriffs auf Daten in Grid-Umgebungen eingesetzt. Seine Funktionalitäten umfassen die dynamische Allokation von Speicherplatz, das Aushandeln von Transportprotokollen sowie das Management von Dateien auf Speicherressourcen.

In auf *gLite* basierenden Grid-Umgebungen werden SRM-Services als Schnittstelle zu den eigentlichen Speicherressourcen verwendet. Jede Datei auf einer Speicherressource ist über die SRM-Schnittstelle mittels einer SURL eindeutig adressierbar. Diese Art der Adressierung wird jedoch nicht von den Nutzern verwendet. Nutzer verwenden die Adressierung anhand des LFN eines Objekts, eine Übersetzung durch den *File Catalog* ist somit bei allen Zugriffen erforderlich.

IO-Service Der IO-Service ist die zentrale Komponente in *gLite* für den Zugriff auf Speicherressourcen. Der Nutzer initiiert Zugriffe auf Objekte, die über ihren LFN angesprochen werden über Befehle, die den UNIX-Kommandos für Dateioperationen nachempfunden sind. Der IO-Service löst den LFN über den *File Catalog* in eine Storage URL auf, dabei wird auch die Zugriffskontrolle durchgeführt. Auch der anschließende Dateitransfer zwischen Nutzer und Speicherressource erfolgt über den IO-Service. Diese Funktionalität erhöht einerseits die Sicherheit, da der Zugriff auf SRM-Ressourcen auf den IO-Service beschränkt werden kann, sie ist jedoch auch ein Engpass, der zu deutlichen Einbußen in der Leistungsfähigkeit des Systems führen kann.

2.1.3.3 Security Services

Die von Grid-Infrastrukturen bereitgestellte Rechenleistung und Speicherkapazität stellt ein interessantes Ziel für missbräuchliche Nutzung dar. Auch Zugriff auf die Daten der Nutzer kann für Angreifer von Wert sein. Aus diesem Grund stellt die Absicherung von Grid-Ressourcen eine wesentliche Aufgabe von Grid-Middleware dar. Die Basisfunktionalitäten sind zum Einen die Authentifizierung (Kapitel 2.2) von Nutzern und Diensten, zum Anderen die Autorisierung von Nutzern (vgl. Kapitel 2.3.4) und eine darauf aufbauende Zugriffskontrolle (vgl. Kapitel 2.3.1). Zusätzlich bietet *gLite* zwei zentrale Komponenten, den *MyProxy Credential Management Service* und den *Virtual Organisation Membership Service* (VOMS).

MyProxy Credential Management Service Der MyProxy-Service [NTW01] bietet im Kontext einer auf *gLite* aufbauenden Grid-Infrastruktur zwei Dienste an. Seine erste Funktionalität ist die sichere Speicherung der Zertifikate und der zugehörigen geheimen Schlüssel der Nutzer (vgl. Kapitel 2.2.2). Der Nutzer hinterlegt diese beiden Dateien auf dem MyProxy-Server und bezieht für die Nutzung der Grid-Infrastruktur ein davon abgeleitetes Proxy-Zertifikat (vgl. Kapitel 2.2.4) mit nur kurzer Gültigkeitsdauer. Die zweite Funktionalität ist die Erneuerung von Proxy-Zertifikaten in Zusammenarbeit mit dem WMS. Dieser als Proxy-Renewal bezeichnete Vorgang ermöglicht Grid-Jobs, die länger aktiv sind als das zum Starten des Auftrags verwendete Proxy-Zertifikat.

Virtual Organisation Membership Service Der VOMS verwaltet die Nutzer einer Virtuellen Organisation (vgl. Kapitel 2.1.1). Diesen Nutzern können Gruppenmitgliedschaften und Rollen innerhalb der VO sowie generische Attribute zugeordnet werden. Diese Eigenschaften werden dem Nutzer in signierter Form durch den VOMS bestätigt und auf den Ressourcen für die Autorisierung von Zugriffen verwendet (vgl. Kapitel 2.3.4).

2.1.3.4 Information and Monitoring Services

Grid-Infrastrukturen haben die Eigenschaft, Ressourcen dynamisch in den Verbund aufnehmen und auch wieder aus dem Verbund entlassen zu können. Eine statische Konfiguration der zur Verfügung stehenden Dienste scheidet aus diesem Grund aus. Informationssysteme, die alle Systeme der Grid-Infrastruktur kennen und über die von ihnen bereitgestellten Dienste Auskunft erteilen können, treten an die Stelle der statischen Konfiguration. Zusätzlich überwachen Monitoring Services den aktuellen Auslastungs- und Betriebszustand von Ressourcen. Die gLite-Middleware stellt zwei Komponenten für die *Information and Monitoring* Services bereit, den *Relational Grid Monitoring Architecture* (R-GMA) Service und den *Logging & Bookkeeping* (LB) Service.

Relational Grid Monitoring Architecture Der R-GMA-Service wird im Rahmen der *gLite Service Discovery* (SD) dazu verwendet, Informationen über die zur Verfügung stehenden Ressourcen im Grid zu sammeln und auf Anfrage auszugeben. Diese Informationen werden von den Ressourcen periodisch an den R-GMA-Service gemeldet und umfassen Verfügbarkeit, Art des Dienstes und Auslastungszustand.

Logging & Bookkeeping Service In Grid-Infrastrukturen kann simultan eine große Anzahl von Rechenaufträgen aktiv sein, an deren Bearbeitung jeweils mehrere Dienste beteiligt sind. Im Fehlerfall müssen alle beteiligten Komponenten untersucht werden, um die Ursache für einen Abbruch des Auftrags zu finden. Der LB-Service vereinfacht diesen Vorgang, indem jedem Grid-Job eine eindeutige Identifikation zugeordnet wird, unter der alle an der Bearbeitung beteiligten Dienste Statusmeldungen auf diesem ablegen. Die auf dem LB-Service abgelegten Informationen sind durch externe Zugriffe nicht veränderbar, es können lediglich Informationen zu einem Auftrag hinzugefügt werden. Der LB-Service stellt eine zentrale Instanz für die Aufnahme von Logging-Informationen in Grid-Infrastrukturen auf Basis der gLite-Middleware dar.

2.2 Authentifizierung im Grid-Computing

Authentifizierung ist definiert als die Überprüfung der behaupteten Identität eines Kommunikationspartners. In verteilten Systemen werden Authentifizierungsmechanismen eingesetzt, wenn die Identität eines Nutzers möglichst zweifelsfrei festgestellt werden muss. Dies ist bei allen Systemen und Anwendungsgebieten mit erhöhtem Schutzbedarf der Fall, so z. B. wenn sensitive oder personalisierte Daten verwaltet werden oder die Nutzung des Systems mit signifikanten Kosten verbunden ist.

Durch den Aufbau von Grid-Verbünden wird angestrebt, Ressourcen einer Anzahl von Institutionen einer Nutzerschaft zugänglich zu machen. Dabei kommt eine anonyme Nutzung von Grid-Ressourcen aus verschiedenen Gründen nicht in Betracht:

- Die Mittel für den Aufbau der Infrastruktur werden in der Regel zweckgebunden, d. h. für den Nutzen einer bestimmten Anwendergruppe bereitgestellt. Die Anwender müssen folglich für die Nutzung der Ressourcen autorisiert sein. Eine Autorisierung ist jedoch nur nach erfolgter Authentifizierung möglich (vgl. Kapitel 2.3).
- Der Zugriff auf Daten Dritter, die auf Grid-Ressourcen gespeichert werden, muss bei Anwendungsgebieten mit einem erhöhten Bedarf an Datensicherheit unterbunden werden können.
- Es muss nachvollziehbar sein, wer eine Handlung auf dem Grid ausgeführt hat. Diese Funktionalität ist insbesondere für die Abrechnung erforderlich, jedoch auch für die Verfolgung missbräuchlicher Nutzung.

Neben der sicheren Authentifizierung von Nutzern im Grid ist auch der gegenseitige Nachweis der Identität zwischen Nutzer und Grid-Ressource sowie zwischen Grid-Ressourcen erforderlich. Letztere Funktionalität wird von allen Grid-Ressourcen verwendet, ist jedoch insbesondere im Falle von Ressourcen notwendig, die besondere Vertrauensstellungen im Grid innehaben, wie Dienste, die die Rechte von Nutzern verwalten.

Mit der Feststellung, dass eine Authentifizierungsfunktionalität für Nutzer im Grid sowie dessen Ressourcen benötigt werden, stellt sich die Frage nach dem zu verwendenden Verfahren. Das einfachste Verfahren zum Nachweis der eigenen Identität ist die Authentifizierung mittels einer Username/Passwort-Kombination. Dieses Verfahren ist jedoch für einen sicheren Betrieb eines Grid nicht anwendbar. Einerseits gilt es als unsicher, da Nutzer dazu tendieren, zu kurze oder auf Worten oder Namen basierende Passwörter zu wählen und diese nicht oder nur sehr selten zu ändern. Auf der anderen Seite ist die Verwaltung einer konsistenten Passwort-Datenbank über viele Institutionen hinweg fehlerträchtig. Die meisten heute verwendeten Grid-Umgebungen setzen daher Authentifizierungsmechanismen ein, die auf asymmetrischer Kryptographie und Public-Key-Infrastrukturen (PKI) basieren. Diese Verfahren gelten mit entsprechend gewählten Schlüssellängen als sicher. Auch skalieren sie in verteilten Umgebungen mit einer großen Nutzerschaft und vielen beteiligten

Institutionen gut und benötigen anders als vergleichbare Verfahren wie Kerberos [SNS88] keine stets erreichbaren Dienste.

In den nächsten Abschnitten wird ein Überblick über Public-Key-Kryptographie und Public-Key-Infrastrukturen gegeben, bevor auf die Grid-spezifischen Mechanismen in Kapitel 2.2.3 eingegangen wird.

2.2.1 Public-Key-Kryptographie

Kryptographie dient der Verschlüsselung von Nachrichten mittels eines krypthographischen Algorithmus und einem oder mehreren Schlüsseln. In der Regel wird heute zwischen symmetrischen und asymmetrischen Kryptosystemen unterschieden. Symmetrische Verfahren (wie z. B. der AES) verwenden den gleichen Schlüssel zur Ver- und Entschlüsselung der Nachricht. Bei diesen Verfahren besteht eine grundsätzliche Schwierigkeit darin, den Schlüssel zu einem Chiffretext dem Empfänger der Nachricht in der Art mitzuteilen, dass ihn kein Dritter abhören kann. Um das Problem des Schlüsselaustausches zu lösen, wurden asymmetrische oder auch Public-Key-Algorithmen entwickelt. Die grundlegenden Prinzipien dieser neuen Klasse von kryptographischen Algorithmen wurden 1976 von Whitfield Diffie und Martin E. Hellman auf der *National Computer Conference* [DH76a] und später in dem Artikel *New Directions in Cryptography* [DH76b] vorgestellt. Die Basis asymmetrischer Verschlüsselungsverfahren ist die Verwendung unterschiedlicher, aber verwandter Schlüssel für Chiffrierung und Dechiffrierung. Dabei muss es prinzipiell oder durch einen zu hohen Aufwand unmöglich sein, aus einem Schlüssel den anderen zu berechnen. Ein bekanntes Public-Key-Verfahren ist das von Ron Rivest, Adi Shamir und Leonard Adleman entwickelte RSA.

Anwendung finden Public-Key-Kryptosysteme hauptsächlich beim Austausch von Schlüsseln für symmetrische Verschlüsselungsverfahren, in Signaturverfahren für digitale Dokumente oder in Authentifizierungssystemen. Für die Verschlüsselung von Nachrichten werden in der Regel symmetrische Verfahren eingesetzt, da diese deutlich weniger Rechenschritte für die gleiche Menge an Nachrichtentext benötigen und dadurch um mehrere Größenordnungen schneller sind.

2.2.1.1 Verschlüsselung von Nachrichten

Jeder Teilnehmer einer bidirektionalen Kommunikation generiert sich vor deren Beginn ein Schlüsselpaar, dass aus einem geheimen (*private*) und einem öffentlichen (*public*) Schlüssel besteht. Nachfolgend kann der öffentliche Schlüssel in einem öffentlichen Verzeichnis den anderen Kommunikationspartnern zur Verfügung gestellt werden. In Abbildung 2.4 auf der nächsten Seite ist der schematische Ablauf der Verschlüsselung einer Nachricht beim Sender (Alice) und nachfolgender Entschlüsselung beim Empfänger (Bob) abgebildet. Es wird davon ausgegangen, dass

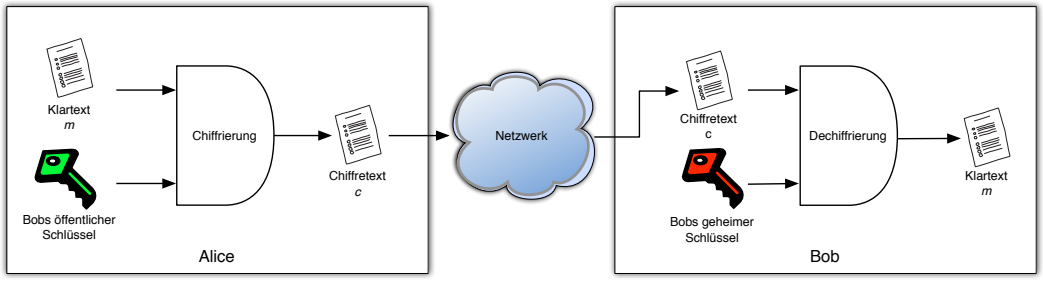


Abbildung 2.4: Verschlüsselung von Nachrichten mittels Public-Key-Kryptographie

Alice bereits vor Beginn der Kommunikation über den öffentlichen Schlüssel von Bob verfügt, dass sie ihn z. B. aus einem öffentlichen Verzeichnis bezogen hat. Im ersten Schritt bildet Alice den Chiffretext (c) aus der Klartext-Nachricht (m) mittels der Verschlüsselungsfunktion (E) und dem öffentlichen Schlüssel von Bob (k_{pB}):

$$c = E_{k_{pB}}(m)$$

Der Chiffretext kann nun über das als unsicher eingestufte Netzwerk übertragen werden. Auf der Empfängerseite stellt Bob die ursprüngliche Nachricht mittels seines geheimen Schlüssels (k_{sB}) und der Entschlüsselungsfunktion (D) wieder her:

$$m = D_{k_{sB}}(c)$$

2.2.1.2 Digitale Signatur von Nachrichten

Die digitale Signatur von Nachrichten mittels Public-Key-Kryptographie dient der Überprüfung der Integrität und des Ursprungs einer Nachricht. Mit der digitalen Signatur kann der Empfänger einer Nachricht mit dem öffentlichen Schlüssel des (vermeintlichen) Absenders überprüfen, ob die Nachricht durch den geheimen Schlüssel des Senders signiert wurde und ob sie auf dem Weg zum Empfänger verändert wurde.

Der Ablauf einer digitalen Signatur einer Nachricht ist in Abbildung 2.5 auf der nächsten Seite dargestellt. Es ist dem Verfahren der Verschlüsselung ähnlich, differiert jedoch insbesondere in den verwendeten Schlüsseln.

Will Alice eine Nachricht an Bob verfassen und diese digital signieren, erzeugt sie mittels einer Einweg-Hashfunktion einen Hashwert (h) aus der Klartext-Nachricht (m). Dieser Hashwert wird nun von Alice unter Verwendung ihres geheimen Schlüssels (k_{sA}) signiert:

$$sig = S_{k_{sA}}(h(m))$$

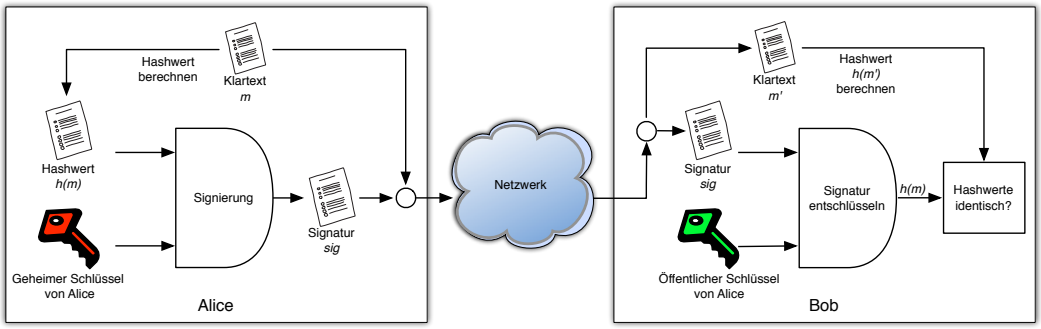


Abbildung 2.5: Digitale Signatur von Nachrichten mittels Public-Key-Kryptographie

Die Signatur wird nun zusammen mit der Klartext-Nachricht an Bob gesendet. Dieser wendet nun die gleiche Hashfunktion auf den erhaltenen Klartext (m') an. Nachfolgend kann er mit dem zum Signaturalgorithmus passenden Verifizierungsalgorithmus und dem öffentlichen Schlüssel von Alice die erhaltene Signatur entschlüsseln:

$$V_{k_{pA}}(sig) = h(m)$$

Ist das Ergebnis der entschlüsselten Signatur $h(m)$ mit dem aus der Nachricht berechneten Hashwert $h(m')$ identisch, wurde die Nachricht mit dem geheimen Schlüssel von Alice signiert und auf dem Weg zu Bob nicht verändert. Unter der Annahme, dass der geheime Schlüssel von Alice nicht kompromittiert wurde steht nun fest, dass Alice die Nachricht an Bob signiert hat.

2.2.1.3 Authentifizierung mittels asymmetrischer Verschlüsselung

Die digitale Signatur von Nachrichten ist eine Methode um nachzuweisen, dass eine Nachricht von einem bestimmten Teilnehmer einer Kommunikation stammt. Ein ähnliches Verfahren kann für Authentifizierungszwecke angewendet werden. Dabei erstellt der Partner (Bob), bei dem sich Alice authentifizieren will, eine Zufallszahl r (vgl. Abbildung 2.6 auf der nächsten Seite) und sendet sie an Alice. Alice signiert die Zahl mit ihrem geheimen Schlüssel und sendet r_{sig} an Bob, der die Signatur mit Hilfe des öffentlichen Schlüssels von Alice überprüfen kann. Ist das Ergebnis r' und die ursprüngliche Zufallszahl r identisch, ist die Partei, die behauptet Alice zu sein, im Besitz des geheimen Schlüssels zum öffentlichen Schlüssel von Alice. Wenn der geheime Schlüssel nicht kompromittiert wurde, ist diese Partei somit Alice.

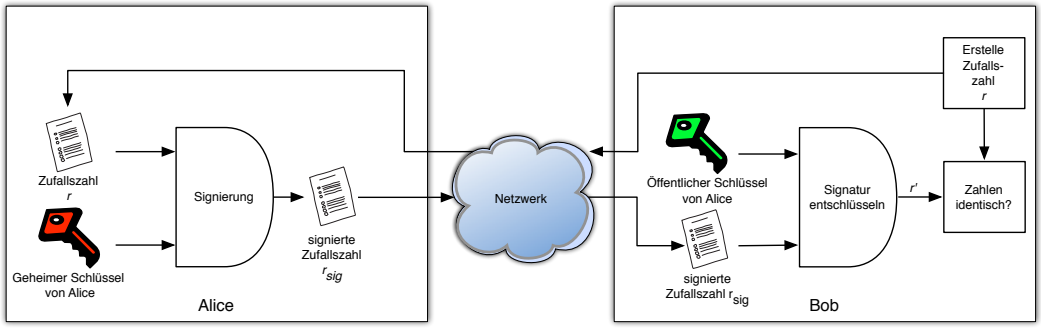


Abbildung 2.6: Authentifizierung von Entitäten mittels Public-Key-Kryptographie

2.2.2 Public-Key-Infrastrukturen

In Kapitel 2.2.1 wurde asymmetrische Kryptographie zur digitalen Signatur und Authentifizierung von Entitäten vorgestellt. Voraussetzung dafür, dass diese Verfahren für die genannten Zwecke verwendet werden können, ist das Vertrauen in die Bindung des öffentlichen Schlüssels eines Teilnehmers an dessen Identität. Dieses Vertrauen kann beispielsweise durch ein direktes Treffen der beteiligten Personen etabliert werden, bei dem die öffentlichen Schlüssel ausgetauscht werden und amtliche Lichtbildausweise überprüft werden. Ein solcher Vorgang ist jedoch in der Regel unwirtschaftlich und nur dann zu rechtfertigen, wenn die Sicherheitsbedürfnisse besonders hoch sind. Um diesen direkten Kontakt zu umgehen, wurden Public-Key-Infrastrukturen entwickelt. Diese ersetzen das direkte Vertrauen der einzelnen Teilnehmer durch ein indirektes Vertrauen in einen Dritten, der die Identität der Teilnehmer zertifiziert.

2.2.2.1 Certification Authorities als Trusted Third-Party

Die Rolle des vertrauenswürdigen Dritten (Trusted Third-Party) übernimmt in einer PKI die *Certification Authority* (CA). Eine CA, die im deutschen Sprachgebrauch als Zertifizierungsstelle bezeichnet wird, verfügt über ein eigenes Schlüsselpaar aus einem geheimen und einem öffentlichen Schlüssel, mit dem sie Zertifizierungen durchführt.

Ein Teilnehmer, der sich den Besitz seines öffentlichen Schlüssels von der CA zertifizieren lassen möchte, muss in einem ersten Schritt seine Identität einer der CA zugeordneten *Registration Authority* (RA) nachweisen. Nachfolgend muss er den Besitz des geheimen Schlüssels zu dem öffentlichen Schlüssel beweisen, den er der RA vorlegt. Wenn die Überprüfung der Identität des Teilnehmers erfolgreich ist, erstellt die CA ein Zertifikat (vgl. Kapitel 2.2.2.2) mit den Daten des Nutzers sowie seinem öffentlichen Schlüssel. Dieses Zertifikat wird von der CA abschließend mit

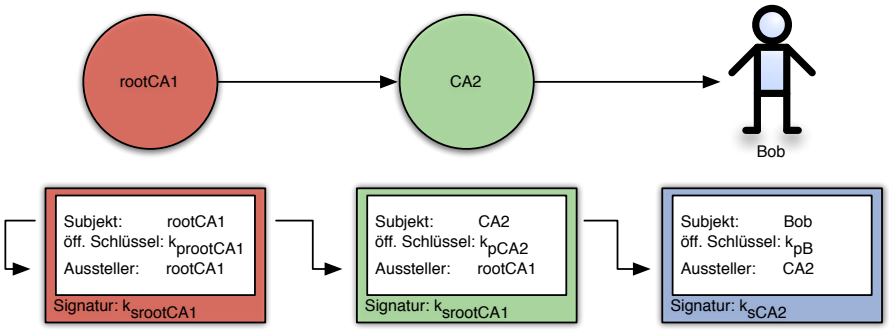


Abbildung 2.7: Zertifizierung von Entitäten in Public-Key-Infrastrukturen

ihrem geheimen Schlüssel signiert. Um ein Zertifikat verifizieren zu können, muss ein Teilnehmer über den öffentlichen Schlüssel der Zertifizierungsstelle verfügen, der im Zertifikat der CA abgelegt ist. Zertifikate von Zertifizierungsstellen sind jedoch im Gegensatz zu den Teilnehmerzertifikaten (*end-entity certificates*) selbst-signiert (*self-signed*), d. h. mit dem geheimen Schlüssel signiert, der zu dem öffentlichen Schlüssel im Zertifikat gehört. Die CA bestätigt sich ihre Identität und den Besitz des Schlüsselpaares also selbst.

Aus organisatorischen Gründen kann jedoch die Hierarchisierung von Zertifizierungsstellen sinnvoll sein. In diesem Fall kommt es vor, dass CA-Zertifikate nicht selbst-signiert sind. Diese Hierarchiebildung erfolgt, indem eine höherstehende CA der nächst niedrigeren ein Zertifikat ausgestellt, das sie aber im Gegensatz zu Teilnehmerzertifikaten dazu berechtigt, weitere Zertifikate auszustellen. In Abbildung 2.7 ist eine hierarchische Struktur von Zertifizierungsstellen und den zugehörigen Zertifikaten dargestellt.

2.2.2.2 Public-Key-Zertifikate nach ITU-T X.509

Die Bindung eines öffentlichen Schlüssels an die Identität einer Entität muss durch alle Kommunikationspartner dieser Entität verifizierbar sein. Um diese Verifizierbarkeit zu gewährleisten, wurde 1997 von der ITU-T die Empfehlung X.509 verabschiedet. In dieser wird ein vereinheitlichtes Format [Int05] für Public-Key-Zertifikate definiert. X.509 wurde 2002 in der Version 3 zusammen mit einem Format für Zertifikatssperrlisten (vgl. Kapitel 2.2.2.3) von der IETF übernommen und in RFC 3280 [HPFS02] veröffentlicht.

X.509-Zertifikate haben eine Anzahl von fest definierten Feldern, können aber durch Erweiterungen flexibel ergänzt werden. Die Relevanz dieser Erweiterungen kann mit der Markierung als *critical* oder *non-critical* festgelegt werden. Ist die Erwei-

terung als *critical* markiert, muss der Empfänger eines solchen Zertifikates dieses zurückweisen, wenn er die Erweiterung nicht unterstützt. Im Falle einer als *non-critical* gekennzeichneten Erweiterung darf der Empfänger frei entscheiden, das Zertifikat zurückzuweisen oder es zu akzeptieren. Um verschiedene Zertifikatserweiterung unterscheiden zu können, werden diese mit einem eindeutigen Bezeichner, dem *Object Identifier* (OID) gekennzeichnet. OIDs sind in einem hierarchischen Namensraum organisiert und in der ITU-T Empfehlung X.680 [Int02] definiert.

In der Folge soll eine kurze Erklärung für die Bedeutung der einzelnen Felder von X.509-Zertifikaten gegeben werden, da der Aufbau dieser Zertifikate in den Kapiteln 4 und 5 dieser Arbeit von Bedeutung ist:

- **Version**. Version, nach der das Zertifikat erstellt wurde. Dieses Feld hat nach RFC 3280 mindestens den Wert 2, heute in der Regel jedoch den Wert 3.
- **Seriennummer**. Die Seriennummer eines Zertifikats wird dafür verwendet, Zertifikate zurückzurufen.
- **Aussteller**. *Distinguished Name* (DN) der ausstellenden Zertifizierungsstelle.
- **Gültigkeitsdauer**. Zeitraum von dem an und bis zu dem das Zertifikat gültig ist, wenn es nicht vorher zurückgerufen wird. Der Gültigkeitszeitraum wird in UTC angegeben.
- **Subjekt**. Entität für die das Zertifikat ausgestellt wurde.
- **Algorithmus des öffentlichen Schlüssels**. Algorithmus, der für die Erstellung des Schlüsselpaares des Subjekts verwendet wurde.
- **öffentlicher Schlüssel**. Der öffentliche Schlüssel des Subjekts.
- **Erweiterungen**. Zusätzliche Informationen. Diese können als *critical* oder *non-critical* markiert werden.
- **Signaturalgorithmus**. Für die Signatur verwendeter Algorithmus.
- **Signatur**. Die Signatur der Zertifizierungsstelle, mit der das Zertifikat beglaubigt wird.

2.2.2.3 Rückruf von Public-Key-Zertifikaten nach ITU-T X.509

Zertifizierungsstellen versehen jedes X.509-Zertifikat mit einem Gültigkeitszeitraum, nach dessen Ablauf ein Zertifikat als ungültig anzusehen ist. Im Normalfall sind daher weitere Maßnahmen unnötig, da End-Entity-Zertifikate (EEZ) in der Regel nur für den Zeitraum eines Jahres oder wenig länger gültig sind. Es sind jedoch Situationen denkbar, in denen ein Zertifikat vorzeitig für ungültig erklärt werden muss. Dazu gehören alle Fälle, in denen entweder die Sicherheit des geheimen Schlüssels nicht mehr gewährleistet ist oder sich die Zugehörigkeit eines Zertifikatinhabers zu der Institution, in deren Kontext das Zertifikat ausgestellt wurde, verändert hat, so z. B.

- Verlust des dechiffrierten geheimen Schlüssels
- Ausscheiden des Zertifikatinhabers aus der Institution, die für ihn das Zertifikat beantragt hat

Um Missbrauch beim Widerruf [1] von Zertifikaten zu vermeiden, muss gewährleistet sein, dass der Vorgang nachweisbar von einer dazu befugten Autorität durchgeführt wird. Diese Aufgabe wird im PKI-Kontext meistens von der ausstellenden Autorität, also der Zertifizierungsstelle, wahrgenommen. RFC 3280 [HPFS02] sieht jedoch explizit vor, dass die CA diese Aufgabe an eine andere Autorität delegieren kann.

Wichtig für die Verwendbarkeit von Sperrinformationen ist zusätzlich zu der Information, welche Zertifikate zurückgerufen wurden auch der jeweilige Zeitpunkt ihrer Sperrung. Handlungen, die der rechtmäßige Besitzer des Zertifikats vor diesem Zeitpunkt mit dem Zertifikat und dem geheimen Schlüssel ausgeführt hat, fallen weiterhin in seinen Verantwortungsbereich und können nicht durch ihn abgestritten werden.

Gebräuchlich sind zwei Mechanismen zum Widerrufen von Zertifikaten. Dies sind Sperrlisten, die auch als *Certificate Revocation Lists* (CRL) bezeichnet werden sowie das in RFC 2560 [MAM+99] spezifizierte *Online Certificate Status Protocol* (OCSP). Beide Verfahren werden im Folgenden kurz vorgestellt.

CRL Sperrlisten nach Version 2 des X.509-CRL-Profils werden wie auch X.509-Zertifikate in RFC 3280 spezifiziert. Eine Sperrliste wird von der CA periodisch aktualisiert und enthält im Header der Datei neben der obligatorischen Versionsnummer den Aussteller der CRL und den verwendeten Signaturalgorithmus. Der Header enthält weiterhin den Zeitpunkt der Erstellung der Sperrliste sowie den Zeitpunkt der nächsten Aktualisierung. Die nächste Aktualisierung (*next Update*) gibt den spätesten Zeitpunkt an, an dem diese CRL ihre Gültigkeit verliert. Die Aktualisierung kann jedoch bei Bedarf auch früher erfolgen. Die Nutzdaten der Sperrliste bestehen aus den Seriennummern der Zertifikate, die gesperrt werden. Zu jeder Seriennummer wird ein Zeitstempel vermerkt, der den Zeitpunkt des Rückrufs angibt. Abschließend folgt die Signatur über die Sperrliste. Optional können in der Version 2 des X.509-CRL-Profils zusätzliche Erweiterungen eingefügt werden, wie dies auch in X.509-Zertifikaten möglich ist.

OCSP Das OCSP ist im Gegensatz zu den CRL ein Kommunikationsprotokoll und wurde dafür ausgelegt, Informationen über den momentanen Status eines Zertifikats zu transportieren. Anfragen bezüglich des Status eines Zertifikats richtet der Klient an den OCSP-Responder. In der Anfrage spezifiziert der Klient neben der Versionsnummer des Protokolls den Typ der Anfrage sowie das Zertifikat, über

[1] Der Widerruf von Zertifikaten wird auch als Sperrung bezeichnet.

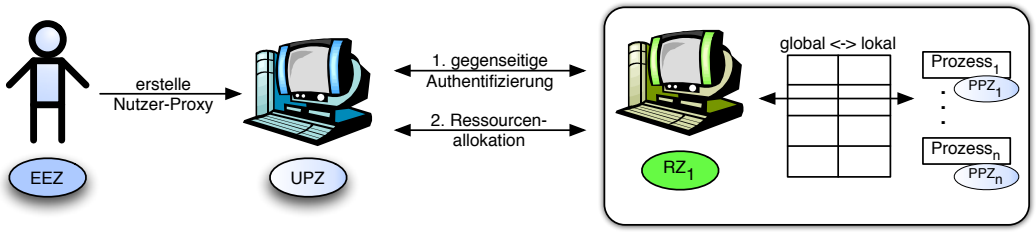


Abbildung 2.8: Basisoperationen der GSI

das er Auskunft anfordert. Dieser antwortet mit dem momentanen Status des Zertifikats. Der Status kann die Werte *good*, *revoked* oder *unknown* annehmen. Über die Antworten des OCSP-Responders wird ein Hashwert gebildet und signiert und als Teil der Antwort an den Klienten gesendet, damit dieser die Integrität und Autorität der Nachricht überprüfen kann.

2.2.3 Die Grid Security Infrastructure

Grid-Verbünde stellen besondere Anforderungen an Authentifizierungs- und Autorisierungsmechanismen. Dies ist durch den Umstand begründet, dass sie über signifikante Ressourcen verfügen, wodurch sie zu einem interessanten Angriffsziel werden und zudem über Grenzen von Sicherheitsdomänen hinweg aufgebaut sind. Ein Sicherheitsvorfall bei einem Partner im Grid darf nicht zu einer Kettenreaktion durch die gesamte Infrastruktur führen.

Grid-Umgebungen, die auf GT 2 basieren, verwenden die *Grid Security Infrastructure* [FKTT98] für die Aufgaben der Authentifizierung und Autorisierung von Nutzern und Ressourcen. Die GSI wurde im Rahmen des Globus Projekts entwickelt und wird in weiterentwickelter Form auch in neueren Versionen des *Globus Toolkit* eingesetzt. Weiterhin basieren die Authentifizierungs- und Autorisierungsmechanismen von LCG und *gLite* auf der GSI. Bei der Entwicklung der GSI wurden folgende Ziele verfolgt:

1. Unterstützung des *Single Sign-On* (SSO) für Nutzer
2. Gewährleistung der Sicherheit in einer dynamischen, sich über Domänengrenzen erstreckenden Infrastruktur
3. Bereitstellung von Mechanismen, die eine authentifizierte und vertrauliche Kommunikation zwischen Grid-Ressourcen ermöglichen
4. Implementierung von Delegationsmechanismen

Wie in der GSI die genannten Ziele umgesetzt werden, soll anhand von Abbildung 2.8 auf der vorherigen Seite erläutert werden. In der Grafik sind die Basisoperationen und -konzepte der GSI dargestellt.

Das erste Ziel, die Unterstützung von *Single Sign-On* (SSO) für Nutzer des Grid, soll diesem wiederholte Authentifizierungsvorgänge, in der Praxis also eine wiederholte Eingabe eines Passwortes oder einer Passphrase ersparen. Die GSI sieht hierfür einen Stellvertreter für den Nutzer vor, der als *User Proxy* bezeichnet wird. Dieser wird durch den Nutzer eingerichtet und agiert für die Laufzeit des Auftrags stellvertretend für den Nutzer auf seinem lokalen Rechner[2]. Foster et al. definieren den *User Proxy* in [FKTT98]:

> „A user proxy is a session manager process given permission to act on behalf of a user for a limited period of time."

Für den Zugriff auf Ressourcen im Namen des Nutzers benötigt der *User Proxy* Zugriff auf dessen Berechtigungsnachweise (*Credentials*), die in der Grafik durch End-Entity-Zertifikate repräsentiert werden. Die GSI verfügt hierfür über Mechanismen, die eine zeitlich beschränkte Ableitung von den Credentials des Nutzers ermöglichen. In der Praxis werden für diese Ableitung Proxy-Zertifikate (vgl. Kapitel 2.2.4) verwendet, die hier als *User Proxy Zertifikate* (UPZ) bezeichnet werden.

Das zweite Ziel, die Gewährleistung der Sicherheit über Domänengrenzen hinweg, soll über den *Resource Proxy* genannten Mechanismus erreicht werden. Foster et al. definieren diesen wie folgt:

> „A resource proxy is an agent used to translate between interdomain security operations and local intradomain mechanisms."

Der *Resource Proxy* dient folglich als ein Übergang zwischen dem Grid und der lokalen Ressource. Dabei kommt ihm auch die Aufgabe zu, die Abbildung von *Global Subject* auf *Local Subject*, also die Abbildung von *Distinguished Name* auf ein lokales Nutzerkonto vorzunehmen. Im *Globus Toolkit* übernimmt die Aufgabe des *Resource Proxy* der GRAM (vgl. Kapitel 2.1.2). Dieser tritt als eigenständige Entität im Grid auf, verfügt daher über eigene Credentials (RZ) zur Authentifizierung.

Das dritte Ziel ist die Bereitstellung von Mechanismen, die eine authentifizierte und vertrauliche Kommunikation zwischen Grid-Ressourcen ermöglichen. Dieses Ziel realisiert die GSI über die Nutzung bekannter Mechanismen und Komponenten wie Public-Key-Kryptographie und die Verwendung der GSS-API [Lin00]. Diese wird genutzt, um Protokollabläufe wie die Signatur von Daten oder die gegenseitige Authentifizierung von Nutzer und Ressource von den genutzten Sicherheitsmechanismen wie SSL/TLS [DA99] oder Kerberos [SNS88] zu entkoppeln. Im Rahmen der Arbeiten am *Globus Toolkit* wurde die GSS-API signifikant um Fähigkeiten zum Im- bzw. Export von Credentials und zum Umgang mit Erweiterungen von Proxy-Zertifikaten erweitert [MWTE04]. Auch wenn die GSI auf der GSS-API aufsetzt,

[2] Dieser Rechner wird in der Regel als *Local Host* oder *User Interface* bezeichnet.

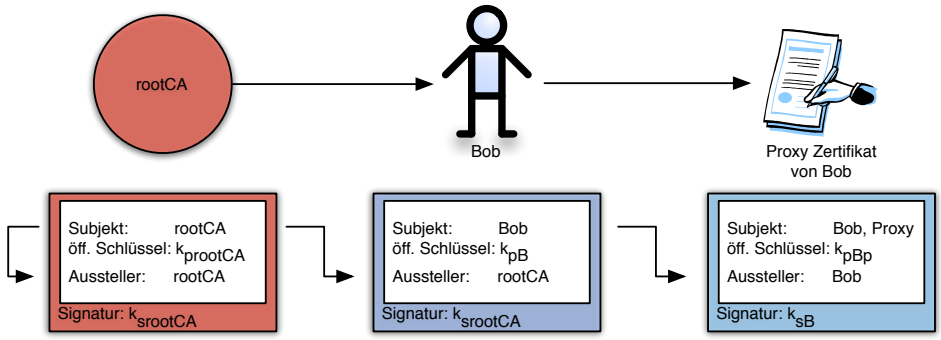


Abbildung 2.9: Ableitung eines Proxy-Zertifikates von einem End-Entity-Zertifikat

um eine Entkopplung von den genutzten Sicherheitsmechanismen zu erreichen, verwenden die meisten Grid-Verbünde heute auf Public-Key-Verfahren basierende Authentifizierungsmechanismen.

Das vierte Ziel, die Implementierung von Delegationsmechanismen, soll Ressourcen, die den Auftrag eines Nutzers bearbeiten in die Lage versetzen, in dessen Namen zu agieren (vgl. Kapitel 2.3.3). So soll es im Rahmen eines Auftrags möglich sein, Dateien von einem Speichersystem zu beziehen oder auch weitere Aufträge auf anderen Ressourcen zu starten. Zu diesem Zweck werden den Prozessen über SSL/TLS Credentials (PPZ_n) delegiert, die vom EEZ des Nutzers abgeleitet wurden. Diese haben ebenso wie das UPZ eine beschränkte Gültigkeit, jedoch können die Prozesse alle Privilegien des Nutzers ausüben. Eine Beschränkung der delegierten Privilegien ist in der GSI nicht implementiert.

2.2.4 Proxy-Zertifikate

Die GSI unterstützt – wie in Kapitel 2.2.3 beschrieben – eine *Single Sign-On*-Funktionalität für Nutzer des Grid, ebenso wie die Delegation von Nutzerrechten an Grid-Ressourcen. Implementiert werden diese Funktionalitäten durch den Einsatz von Proxy-Zertifikaten.

Proxy-Zertifikate sind X.509-Zertifikate mit einer kurzen Gültigkeitsdauer von typischerweise 12 Stunden, die von den End-Entity-Zertifikaten der Nutzer abgeleitet werden. Damit handelt der Nutzer ähnlich wie eine Zertifizierungsstelle, nur dass er lediglich Stellvertreterzertifikate für sich selbst und keine Zertifikate für Dritte ausstellt. Dafür ist in Public-Key-Infrastrukturen der Eintrag *CA:TRUE* in den Basisbeschränkungen[3] der Zertifikaterweiterungen notwendig. Mit diesem Eintrag

[3] X509v3 Basic Constraints

zeigt die ausstellende Zertifizierungsstelle an, dass sie das Recht zum Ausstellen weiterer Zertifikate an den Besitzer dieses Zertifikats delegieren will. Bei End-Entity-Zertifikaten ist jedoch in den Basisbeschränkungen *CA:FALSE* gesetzt. Ein auf diese Weise erstelltes Proxy-Zertifikat verletzt folglich das Gültigkeitsmodell von X.509 und wird folglich außerhalb der GSI nicht anerkannt.

Die Ableitung eines Proxy-Zertifikats von einem EEZ läuft ähnlich der Erstellung eines EEZ ab. Der Nutzer erzeugt ein neues Schlüsselpaar, bindet den öffentlichen Schlüssel in das neue Zertifikat ein und signiert dieses mit dem geheimen Schlüssel seines EEZ. Das Proxy-Zertifikat trägt den DN des Nutzers im Subjekt, das jedoch um den Eintrag *CN=proxy* erweitert ist (vgl. Abbildung 2.9 auf der vorherigen Seite). Der zugehörige geheime Schlüssel verbleibt unverschlüsselt und wird typischerweise nur durch die Zugriffsrechte des Dateisystems auf dem lokalen Host geschützt.

Der Einsatz von Proxy-Zertifikaten hat einige Vorteile, so erfolgt die Authentifizierung durch Eingabe der Passphrase, die den geheimen Schlüssel des EEZ entschlüsselt, nur einmal. Nachfolgend können Prozesse für die Gültigkeitsdauer des Proxy-Zertifikats ohne weitere Authentifizierungsvorgänge im Namen des Benutzers gestartet werden. Weiterhin erfordert die Integration in die bestehende PKI nur geringfügige Änderungen am Zertifikatsgültigkeitsmodell. Zudem ist der zu erwartende Schaden im Falle der Kompromittierung des geheimen Schlüssels geringer als bei Kompromittierung des EEZ, da Proxy-Zertifikate nur eine Gültigkeit von wenigen Stunden haben.

Mit dem Einsatz von Proxy-Zertifikaten sind aber auch signifikante Nachteile verbunden, so können z. B. Proxy-Zertifikate nicht zurückgerufen werden. Dies beruht auf dem Umstand, dass Proxy-Zertifikate vom Nutzer ausgestellt werden und nicht von einer Zertifizierungsstelle. Diese können jedoch nur solche Zertifikate mittels Sperrlisten zurückrufen, die sie auch selber ausgestellt haben. Im Fall einer Kompromittierung kann man daher nur das Ende der Gültigkeit des Proxy-Zertifikats abwarten, den Nutzer temporär auf allen Ressourcen durch den Einsatz von lokalen Bannlisten sperren oder das EEZ des Nutzers durch die CA zurückrufen lassen. Letztere Maßnahme ist jedoch mit erheblichen Konsequenzen für den Nutzer verbunden, da er bis zur Ausstellung eines neuen Zertifikats nicht auf das Grid zugreifen kann. Auch werden Sperrlisten – wie in Kapitel 2.2.2.3 beschrieben – nur periodisch aktualisiert, eine Nutzung des zurückgerufenen EEZ kann folglich noch bis zum regulären Ablauf der Gültigkeit des kompromittierten Proxy-Zertifikats möglich sein.

2.2.4.1 Proxy-Zertifikate nach RFC3820

Die in Kapitel 2.2.4 vorgestellten Proxy-Zertifikate werden als GSI Proxy-Zertifikate bezeichnet und folgen keinem Standard. In RFC 3820 [TWE+04] der IETF wur-

de der Versuch unternommen, eine solche Standardisierung zu erreichen. Die Unterstützung für diesen Typ von Proxy-Zertifikaten ist momentan auf *Globus Toolkit* Version 4 beschränkt. Proxy-Zertifikate nach RFC 3820 verzichten auf die Erweiterung des DN im Subjekt des Zertifikats um den Eintrag *CN=proxy*. Stattdessen wird das Zertifikat um eine Erweiterung vom Typ *ProxyCertInfo* ergänzt. Diese bietet zum Einen die Möglichkeit, die maximale Delegationstiefe[4] des Proxy-Zertifikats zu beschränken (vgl. Kapitel 2.3.3). Weiterhin können unter Verwendung des Feldes *proxyPolicy* Restriktionen bezüglich der Delegation von Rechten beschrieben werden. Das *proxyPolicy* Feld bietet die Möglichkeit, über einen *Object Identifier* (OID) eine Sprache zu spezifizieren[5], in der die Restriktionen formuliert sind sowie ein Feld[6], um eine solche Policy in das Proxy-Zertifikat einzubetten. RFC 3820 sieht jedoch keine eigene Sprache vor, in der diese Restriktionen formuliert werden sollen. Es ist dem Nutzer überlassen, diese Auswahl zu treffen. Der Standard gibt lediglich zwei Werte für das *policyLanguage* Feld vor:

- *id-ppl-inheritAll*. Dieser Eintrag signalisiert, dass das Proxy-Zertifikat unbeschränkt ist. Der Aussteller delegiert mit ihm sämtliche ihm zur Verfügung stehenden Rechte.
- *id-ppl-independent*. Das Proxy-Zertifikat ist unabhängig von den Rechten des Ausstellers. Der Aussteller delegiert keine Rechte.

Ist einer dieser Werte gesetzt, darf das Feld *policy* nicht in der *ProxyCertInfo* Erweiterung des Proxy-Zertifikats enthalten sein.

2.3 Autorisierung im Grid-Computing

Autorisierung bezeichnet den Vorgang der Zuweisung von Privilegien an handelnde Subjekte im Kontext eines Systems sowie die Überprüfung dieser Privilegien durch geeignete Mechanismen vor deren Ausübung. Autorisierungssysteme bauen unmittelbar auf die sichere Überprüfung der Identität eines Dienstnutzers durch Authentifizierungsmechanismen auf. Ohne diese wäre eine missbräuchliche Nutzung des durch den Autorisierungsmechanismus geschützten Systems möglich, da sich ein Angreifer als berechtigter Nutzer des Systems ausgeben könnte.

Subjekte eines Autorisierungssystems können Nutzer oder Dienste eines Systems sein, aber auch Systeme selbst oder Gruppen dieser Entitäten. Objekte bilden die zu schützenden Ressourcen. Ihre Granularität kann von vollständigen Rechnersystemen bis zu einzelnen Dateien oder gar Bestandteilen dieser variieren. Die konkrete Ausgestaltung von Privilegien eines Autorisierungssystems ist abhängig von den Objekten auf die sie sich beziehen. Zugriffsrechte bilden das Bindeglied zwischen Subjekten und Objekten und in Kombination mit diesen vollständige Zugriffsregeln.

[4] *pCPathLenConstraint* Feld
[5] *policyLanguage* Feld
[6] *policy* Feld

Der Zugriff auf Objekte kann sowohl direkt geregelt werden, wie in den folgenden Beispielen:

- Nutzer Alice darf Datei x lesen.
- Nutzer Bob darf Programm y ausführen.

Privilegien können aber auch indirekt über ein beschreibendes Attribut und eine damit verbundene Regel gebildet werden. Einem Nutzer wird beispielsweise eine Rolle auf einem System zugeordnet, der wiederum konkrete Zugriffsrechte zugeordnet werden:

Attribut	:	Nutzer Bob ist Administrator auf System z.
Regel	:	Administratoren von System z dürfen uneingeschränkt auf dieses zugreifen.
Privileg	:	Nutzer Bob darf uneingeschränkt auf System z zugreifen.

Die Autorisierung von Nutzern und Diensten in Grid-Infrastrukturen basiert grundlegend auf etablierten Verfahren und Mechanismen, seien es Zugriffskontrollmechanismen wie *Role Based Access Control* (vgl. Kapitel 2.3.1.3) oder Autorisierungsarchitekturen wie das *Push-Modell* (vgl. Kapitel 2.3.2.2). Die spezielle Problematik der Autorisierung im Grid-Kontext liegt in der Verteilung der Infrastruktur über eine Vielzahl von weitgehend unabhängigen Institutionen. Die Autoren der *Global Security Architecture* [EGE05] des europäischen EGEE-Projekts schreiben zu diesem Problem:

> „The core problem with authorization in a Grid setting is how to handle the overlay of policies from multiple administrative domains (VO specific policy, operational procedures, site-local policy) and how to combine them.“

Zu der Komplexität, die eine verteilte Infrastruktur mit sich bringt, kommt eine heterogene Nutzerschaft, die in Virtuellen Organisationen (vgl. Kapitel 2.1.1) organisiert ist. Jede der realen und Virtuellen Organisationen definiert Richtlinien bezüglich der Nutzung der von ihr zur Verfügung gestellten und ihr zur Verfügung stehenden Systeme. Diese Richtlinien konfliktfrei in einem Autorisierungssystem zur Deckung zu bringen ist eine aktuell erst in Ansätzen gelöste Herausforderung.

2.3.1 Zugriffskontrollmodelle

Die Durchsetzung der Regeln, die ein Autorisierungssystem aufstellt, erfolgt durch die Zugriffskontrolle. Diese soll den Zugriff auf Objekte durch Subjekte verhindern, die über keine ausreichenden Rechte dafür verfügen. Von den in der Literatur beschriebenen Modellen sind die nachfolgend aufgeführten in der Praxis relevant und finden in verteilten Systemen Anwendung.

2.3.1.1 Discretionary Access Control

Zugriffskontrolle nach dem Modell der *Discretionary Access Control* (DAC) ordnet jedes Objekt einem Eigentümer zu. Dieser entscheidet darüber, welchem Subjekt Rechte für den Zugriff auf das Objekt zugeordnet werden. Ein Subjekt kann jede handelnde Entität bilden und nicht nur ein Nutzer des Systems. Als Subjekte können ebenso andere Rechner sowie Dienste oder Prozesse adressiert werden, die auf einem System arbeiten. Objekte bilden alle Ressourcen eines Systems, dies sind z. B. Dateien oder Programme, aber auch Dienste.

Der Eigentümer verwaltet für jedes Objekt eine Matrix aus Subjekten mit zugeordneten Rechten an diesem Objekt. Eine derartige Matrix wird als *Access Control List* (ACL) bezeichnet. Ein Beispiel für eine ACL, die die Zugriffsrechte auf eine Datei verwaltet, ist in Tabelle 2.1 aufgeführt. Durch diese ACL ordnet der Eigentümer Nutzer *Alice* das Recht zu, diese zu lesen (r), während Nutzer *Bob* zusätzlich das Recht erhält, schreibend (w) auf diese zuzugreifen.

Subjekt	Rechte		
	r	w	x
Alice	+	-	-
Bob	+	+	-

Tabelle 2.1: Access Control List für eine Datei

Das DAC-Modell ist wegen seiner einfachen Konfigurierbarkeit sehr verbreitet, Anwendung findet es z. B. in UNIX-Betriebssystemen. Durch den hohen Grad an lokaler Kontrolle über einzelne Objekte durch deren Eigentümer ist es jedoch schwierig, systemweite Richtlinien durchzusetzen.

2.3.1.2 Mandatory Access Control

Im Gegensatz zum DAC-Modell ist die *Mandatory Access Control* (MAC) ein zentral gesteuertes Zugriffskontrollsystem. Alle Subjekte eines Systems bekommen Freigaben (*Clearance*) zugeordnet, die Objekte werden nach Schutzstufen klassifiziert. Anhand der Freigabe des Subjekts wird über den Zugriff auf ein Objekt entschieden. Ein Zugriff ist nur dann gestattet, wenn das Subjekt mindestens über eine gleichrangige Einstufung wie das Objekt verfügt. Anwendung findet diese Art der Zugriffskontrolle vornehmlich im militärischen Bereich. MAC ist in [DOD85] definiert als:

> „A means of restricting access to objects based on the sensitivity (as represented by a label) of the information contained in the objects and the

> formal authorization (i.e., clearance) of subjects to access information of such sensitivity."

Ein frühes Beispiel für eine MAC, die im Auftrag des US-amerikanischen Militärs entwickelt wurde, ist das Bell-LaPadula Modell [BL76] aus dem Jahr 1976. Bell-LaPadula kombiniert die Klassifizierung[7] von Subjekten und Objekten mit einer zu dieser orthogonalen Unterteilung in Sachbereiche oder Kategorien[8]. Ein Paar aus Klassifizierung und Kategorie ergibt den Sicherheitslevel (*security level*). Um Zugriff auf ein Objekt zu erhalten, muss der Sicherheitslevel des Subjekts den des Objekts dominieren, d. h. die Klassifizierung des Subjekts muss höher als die des Objekts sein und die Kategorieeinstufung des Subjekts muss die des Objekts enthalten.

Zugriffe auf Objekte werden nach Beobachtung (*observation*) und Veränderung (*alteration*) unterschieden. Daraus ergeben sich vier Kombinationen von Zugriffen:

Zugriffsart	Bedeutung
execute (e)	weder Beobachtung noch Veränderung
read (r)	Beobachtung, aber keine Veränderung
append (a)	keine Beobachtung, aber Veränderung
write (w)	sowohl Beobachtung als auch Veränderung

Tabelle 2.2: Kombinationen von Zugriffen im Bell-LaPadula Modell

Bell-LaPadula führt noch zwei weitere Regeln ein, um den Informationsfluss zu regulieren. Damit soll verhindert werden, dass Informationen von einem höheren Sicherheitslevel auf einen niedrigeren weitergereicht werden:

1. *no read-up* oder *simple security property*. Diese Regel besagt, dass kein Subjekt auf Informationen mit einem höheren *security level* zugreifen darf.
2. *no write-down* oder $*$*-property*. Subjekte mit einem höheren Sicherheitslevel dürfen nicht in Dokumente eines niedrigeren Sicherheitslevels schreiben.

Um die Weitergabe von Informationen von einem höheren zu einem niedrigeren Level zu ermöglichen, wird eine Ausnahme von der $*$*-property*-Regel für die dafür eingeführten *trusted subjects* gemacht. Diese vertrauenswürdigen Subjekte sind in [BL76] folgendermaßen definiert:

> „A trusted subject is one guaranteed not to consummate a security-breaching information transfer even if it is possible."

[7] nach [BL76]: *top-secret* > *secret* > *confidential* > *unclassified*.
[8] nach [BL76] ohne Rangfolge: *Nuclear*, *NATO*, *Crypto*

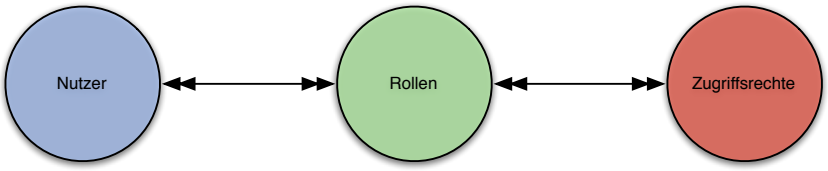


Abbildung 2.10: Entkopplung von Nutzern und Rechten im RBAC-Modell

Die Umsetzung dieser Ausnahmen, d. h. die Erhebung von Subjekten zu *trusted subjects*, ist in der Praxis besonders restriktiv zu handhaben. Ein Bruch dieses besonderen Vertrauens kann die Sicherheit des gesamten Autorisierungssystems gefährden.

Das Bell-LaPadula Modell bietet Vorteile für Bereiche, die stark hierarchisch organisiert sind und in denen Informationsflüsse besonders kontrolliert werden müssen. Nachteilig ist die strenge Formalisierung des Modells, da es für die meisten Anwendungen zu starr ist. Weiterhin erhöhen Prinzipien wie „blindes Schreiben“ auf Objekte höherer Schutzstufe in der Praxis die Fehleranfälligkeit.

2.3.1.3 Role-Based Access Control

Ferraiolo und Kuhn argumentieren in [FK92], dass bessere Zugriffskontrollverfahren als DAC (vgl. Kapitel 2.3.1.1) für kommerzielle und nicht-militärische Regierungsorganisationen existieren. Ihren Ansatz bezeichnen sie als rollenbasierte Zugriffskontrolle (RBAC). RBAC ist seit 2001 ein *Proposed Standard* des NIST [FSG+01].

Die zentrale These, die dem RBAC-Modell zu Grunde liegt, geht davon aus, dass der Nutzer Informationen, auf die er Zugriff hat, in der Regel nicht besitzt. Diese Informationen gehören in der Regel der Organisation, in der der Nutzer Mitglied ist. Ist diese These richtig, sollte er auch nicht über Zugriffsrechte auf diese Informationen entscheiden. Wie in Abbildung 2.10 dargestellt, entkoppeln Ferraiolo und Kuhn im RBAC-Modell die Zugriffsrechte vom Nutzer und übertragen sie auf die Rollen, die er in einer Organisation innehat. Der Nutzer wird nach diesem Modell dynamisch den Rollen zugeordnet, die er in der Organisation gerade ausfüllt. Eine Rolle ist somit vergleichbar mit einer Nutzergruppe, nur dass sie zusätzlich Rechte beinhaltet. Im Wesentlichen behandelt das RBAC-Modell (in [SCFY96] als $RBAC_0$ bezeichnet) folglich zwei Beziehungen:

1. Die Beziehung von Nutzern zu Rollen. Es kann mehrere Nutzer und Rollen in einer auf dem RBAC-Modell basierenden Zugriffskontrolle geben. Nutzer können Mitglied mehrerer Rollen sein, wie auch Rollen mehrere Mitglieder haben können.

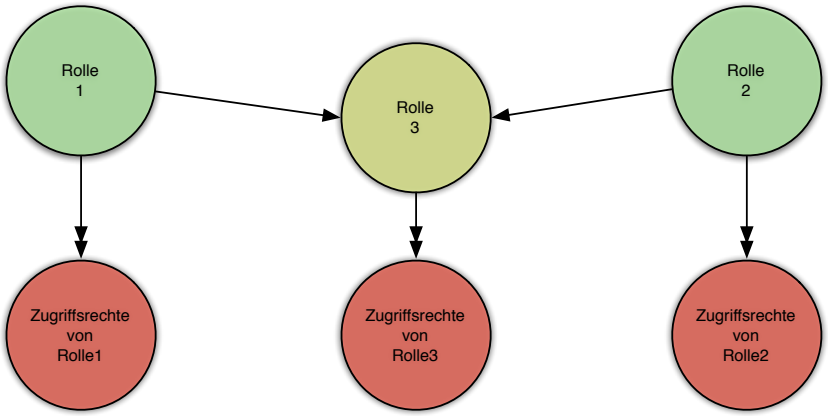


Abbildung 2.11: Hierarchien von Rollen im $RBAC_1$-Modell

2. Die Beziehung von Rollen zu Zugriffsrechten. Eine Rolle kann mehrere atomare Zugriffsrechte enthalten, genauso wie auch einzelne Zugriffsrechte mehreren Rollen zugeordnet sein können.

Ein Nutzer kann im RBAC-Modell Mitglied mehrerer Rollen sein, die er auch zeitgleich aktivieren kann. Es ist aber auch möglich, zwischen den einzelnen Rollen zu wechseln oder zeitgleich mehrere Sitzungen (*sessions*) mit unterschiedlichen aktiven Rollen zu verwenden. Diese Funktionalität kommt der Forderung nach der minimalen Rechtevergabe nach[9].

In [FCK95] und [SCFY96] werden zusätzlich Hierarchien von Rollen eingeführt. In [SCFY96] wird dies als Modell $RBAC_1$ bezeichnet. Hierarchien von Rollen lassen sich dafür nutzen, um Rechte, die von mehreren Rollen benötigt werden, in einer untergeordneten Rolle zusammenzufassen. Die in der Hierarchie höherstehenden Rollen werden dann so definiert, dass sie die Rolle enthalten, der die gemeinschaftlich benötigten Rechte zugeordnet sind (vgl. Abbildung 2.11). Sandhu et. al definieren in [SCFY96] noch zwei weitere RBAC-Modelle:

- $RBAC_2$ führt gegenseitige Ausschlüsse (*constraints*) von Rollen ein. Damit können Rollen definiert werden, deren Mitgliedschaft sich ausschließt. Ein Nutzer kann somit Mitglied von Rolle A, nicht aber von Rolle B und umgekehrt sein. Mit dieser Funktionalität kommt das $RBAC_2$-Modell der Forderung nach der Trennung von Privilegien nach[10].
- $RBAC_3$ beinhaltet sowohl die Bildung von Hierarchien nach $RBAC_1$ als auch von Ausschlüssen nach $RBAC_2$.

[9] *Principle of Least Privilege* [SS75]
[10] *Principle of Separation of Privilege* [SS75]

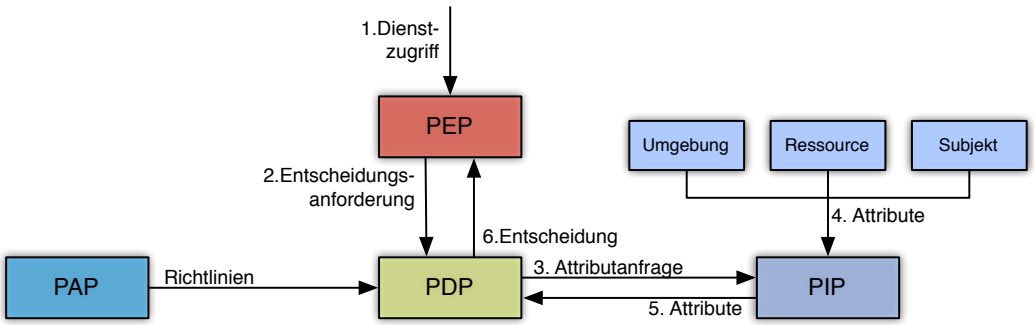


Abbildung 2.12: Autorisierungsmodell des XACML-Standards

Die Vorteile des RBAC-Modells liegen in seiner größeren Flexibilität im Vergleich zum MAC-Verfahren und in seiner besseren administrativen Kontrollierbarkeit gegenüber dem DAC-Modell. Zudem reduziert es die Komplexität erheblich gegenüber DAC, insbesondere wenn die Rollen in einer Organisation relativ stabil sind. In diesem Fall können Nutzer dynamisch und flexibel den Rollen zugeordnet werden, die sie gerade ausfüllen, die Rollen ändern sich dabei nicht.

2.3.2 Autorisierungsarchitekturen

2.3.2.1 Autorisierung anhand von Richtlinien

Die Entscheidung über die Ablehnung oder Gewährung eines Antrags auf Zugang zu einem System erfolgt in der Regel anhand von Richtlinien (*Policies*). Diese Richtlinien werden nach dem Autorisierungsmodell des XACML-Standards [Org05] von spezialisierten Komponenten erstellt bzw. für die Autorisierungsentscheidungen in verteilten Systemen herangezogen. Das Modell ist in Abbildung 2.12 dargestellt, die Komponenten haben folgende Funktionen:

- *Policy Enforcement Point* (PEP). Der PEP fängt den Dienstzugriff des Antragstellers ab. Er hat die Aufgabe, die Entscheidung des PDP durchzusetzen.
- *Policy Decision Point* (PDP). Der PDP entscheidet über Ablehnung oder Gewährung eines Antrags auf Zugang zu einem System. Diese Entscheidung wird anhand von Informationen über den Antragsteller, die Art des gewünschten Zugriffs sowie der Richtlinien des Systems getroffen.
- *Policy Administration Point* (PAP). Der PAP erzeugt die Richtlinien, die vom PDP für die Entscheidung über den Dienstzugriff benötigt werden.
- *Policy Information Point* (PIP). Der PIP stellt Attribute zur Verfügung, die Informationen über Antragsteller, Ressource sowie die Umgebung beinhalten.

2.3.2.2 Kommunikationsmodelle

Wird vor der Nutzung eines Dienstes eine Autorisierung durchgeführt, kann die Kommunikation zwischen Nutzer, Dienst und AAA-Service auf drei Arten [VCF+00] erfolgen. Diese Modelle werden in RFC 2904 als *Agent-*, *Pull-* und *Push-Sequence* bezeichnet. In Grid-Verbünden sind das Pull- und das Push-Modell verbreitet.

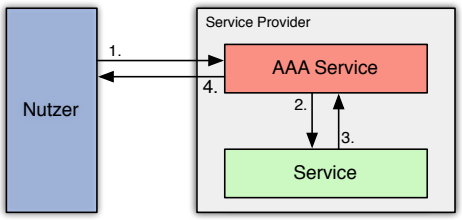


Abbildung 2.13: Agent-Modell nach RFC 2904

Agent-Modell Im Agent-Modell agiert der AAA-Service als Mittler zwischen Nutzer und Service. Der Nutzer kommuniziert während der Initialisierungsphase ausschließlich mit dem AAA-Dienst. Dieser leitet die Dienstanforderung nach erfolgreicher Prüfung der Identität und Autorisierung des Nutzers an den Service weiter. Nachrichten vom Service an den Nutzer werden ebenfalls über den AAA-Service geleitet. In RFC 2904 wird eine direkte Kommunikation zwischen Service und Nutzer nach Abschluss der Initialisierungsphase nicht ausgeschlossen, weitere Untersuchungen werden aber als erforderlich angesehen.

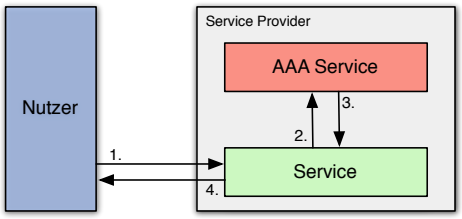


Abbildung 2.14: Pull-Modell nach RFC 2904

Pull-Modell Das Pull-Modell sieht keine Kommunikation des Nutzers mit dem AAA-Service vor. Der zu nutzende Dienst stellt seine Anfrage direkt an den Service, der die Berechtigung nachfolgend beim AAA-Service überprüfen lässt.

Push-Modell Das Push-Modell sieht eine Kommunikation des Nutzers sowohl mit dem AAA-Service als auch mit dem gewünschten Dienst vor. In einer ersten Phase authentifiziert sich der Nutzer beim AAA-Service und fordert eine Autorisierung für den Service an. Bei erfolgreicher Autorisierung stellt dieser dem Nutzer Berechtigungsnachweise für die Nutzung des Dienstes aus. In der zweiten Phase, der Dienstanforderung, präsentiert der Nutzer die Berechtigungsnachweise dem Service, der nach deren Überprüfung den Dienst erbringt.

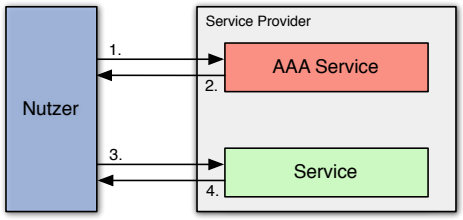


Abbildung 2.15: Push-Modell nach RFC 2904

Das Push-Modell setzt voraus, dass Mechanismen eingesetzt werden, mit denen die Berechtigungsnachweise vor Veränderung auf dem Weg vom AAA-Service über den Nutzer zum Service geschützt werden können. Für diesen Zweck können z. B. Einweg-Hashfunktionen im Verbund mit Verfahren zur digitalen Signatur (vgl. Kapitel 2.2.1.2) eingesetzt werden.

2.3.3 Delegation von Rechten

In verteilten Systemen ist es oftmals erforderlich, mehrere Dienste gleichzeitig oder in zeitlicher Abfolge zu nutzen, um eine Aufgabe auszuführen. Dabei kann es notwendig werden, ein System oder einen Dienst zu ermächtigen, im Namen des aufrufenden Nutzers zu handeln. Dabei muss der Nutzer Rechte, über die er verfügt, an das von ihm ermächtigte System weitergeben können. Ein Anwendungsfall für diese Funktionalität ist gegeben, wenn während der Laufzeit eines durch den Nutzer gestarteten Prozesses, z. B. einer Simulation, auf weitere Dienste wie Dateiserver zugegriffen werden muss und diese Dienste mit Zugriffsbeschränkungen versehen sind. Auch kann es erforderlich sein, Prozesse des Nutzers auf entfernten Systemen zu einem Zeitpunkt auszuführen, zu dem der Nutzer nicht mehr eingeloggt ist. In diesem Fall ist es unmöglich, einen Authentifizierungs- und Autorisierungsvorgang unter aktiver Mitwirkung des Nutzers auszuführen. Die Funktionalität, einen Dritten zu ermächtigen, mit den eigenen Rechten zu handeln, wird als Delegation von Rechten bezeichnet.

Eine frühe Architektur zur Delegation von Rechten [GM90] wurde 1990 von Morrie Gasser und Ellen McDermott vorgestellt. Diese Architektur basiert auf der von

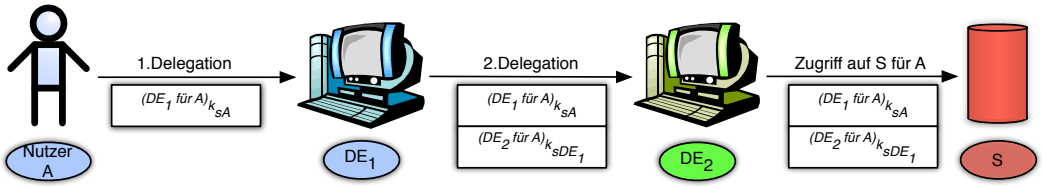


Abbildung 2.16: Delegationskette

der *Digital Equipment Corporation* (DEC) entwickelten *Distributed System Security Architecture* (DSSA) [GGKL89]. In der DSSA werden Verfahren zur Delegation von Rechten unter Zuhilfenahme von asymmetrischer Kryptographie definiert. Der Nutzer stellt dem Delegationsempfänger dabei ein Dokument aus, das als *Delegation Certificate* bezeichnet wird. In diesem wird dem Delegationsempfänger das Recht übertragen, im Namen des Nutzers zu handeln. Das Delegationszertifikat wird vom Nutzer mit seinem geheimen Schlüssel digital signiert, wobei Gasser und McDermott die Verwendung von *Smart Cards* vorsehen[11]. Das Delegationszertifikat wird dann an den Delegationsempfänger weitergeleitet. Dieser kann es nach erfolgreicher Authentifizierung dem aufgerufenen Dienst präsentieren, um Zugriff auf diesen zu erhalten.

Neben dem einfachsten Fall, der Delegation an einen Dritten, können auch verkettete Delegationen durchgeführt werden. Ein solcher Vorgang ist in Abbildung 2.16 dargestellt, dabei ist die in jedem Schritt erforderliche gegenseitige Authentifizierung nicht dargestellt. Der Vorgang läuft in drei Schritten ab:

1. Der Nutzer A meldet sich am System DE_1 an und delegiert seine Rechte an diesen. Dies geschieht, indem er ein Delegationszertifikat erstellt und mit seinem geheimen Schlüssel k_{sA} signiert:

$$(DE_1 \text{ für } A)_{k_{sA}}$$

2. Benötigt der Delegationsempfänger DE_1 die Hilfe eines weiteren Systems (DE_2) für die Bearbeitung der Aufgabe von A, delegiert er dafür die Rechte von A an DE_2 in Form eines weiteren Delegationszertifikats:

$$(DE_2 \text{ für } A)_{k_{sDE_1}}$$

   Er muss jedoch nachweisen, dass er im Besitz der Rechte von A ist. Um dies zu tun, sendet er das Delegationszertifikat von A aus Schritt 1 ebenfalls an DE_2.

[11] Eine Verwendung von digitalen Zertifikaten, die auf anderen Datenträgern gespeichert werden, bedeutet funktional keinen Unterschied. Die Sicherheit des geheimen Schlüssels kann dadurch jedoch beeinflusst werden.

3. Im letzten Schritt kann nun der mit Zugangskontrollen gesicherte Dienst S im Namen von A genutzt werden. Dazu sendet DE_2 die empfangenen Delegationszertifikate zusammen mit dem Dienstaufruf an S.

2.3.3.1 Delegation von Rechten in Grid-Infrastrukturen

In Grid-Infrastrukturen ist die Delegation von Rechten einer der Basismechanismen der Autorisierung. Bedingt durch den Umstand, dass der Nutzer im Normalfall nicht interaktiv auf einer Ressource agiert, ist die Fähigkeit zum Zugriff auf weitere Dienste durch eine im Namen des Nutzers agierende Grid-Ressource zwingend erforderlich. Ein Beispiel für diese Notwendigkeit ist der folgende, sehr einfache Arbeitsablauf:

1. Der Nutzer erteilt einer Grid-Ressource den Auftrag, eine Berechnung durchzuführen. Dazu spezifiziert er in der Beschreibung seines Auftrags das auszuführende Programm sowie die Parameter, mit denen es gestartet werden soll. Weiterhin werden die benötigten Ein- und Ausgabedateien spezifiziert.
2. Die mit der Berechnung beauftragte Grid-Ressource bezieht die für den Auftrag benötigten Eingabedaten von einer Speicherressource. Dies geschieht mit den delegierten Rechten des Nutzers.
3. Die Grid-Ressource führt das Programm des Nutzers aus.
4. Nach Abschluss der Berechnung werden die Ergebnisdateien an eine weitere Speicherressource übertragen. Auch dies geschieht mit den delegierten Rechten des Nutzers.

In GSI-basierten Grid-Middlewares, wie dem *Globus Toolkit* oder *gLite*, tritt an die Stelle des in [GM90] definierten Delegationszertifikats das Proxy-Zertifikat (vgl. Kapitel 2.2.4). Im Gegensatz zum Delegationszertifikat ist es nicht nur ein signiertes Ticket, mit dem Informationen zur Autorisierung übertragen werden, sondern es wird zusammen mit dem geheimen Schlüssel auch zur Authentifizierung eingesetzt. Dabei nimmt der die Delegation erhaltene Dienst die Identität des Nutzers an (Personifikation), die grundlegende Funktionalität der Delegation von Rechten ist jedoch prinzipiell vergleichbar.

Die Delegation (vgl. Abbildung 2.17 auf der nächsten Seite und [WFK+04]) erfolgt durch Ableitung des Proxy-Zertifikats von einem EEC oder einem bereits bestehenden Proxy-Zertifikat über eine gegenseitig authentifizierte Netzwerkverbindung. Die Vertraulichkeit der Verbindung ist jedoch nicht in jedem Fall gewährleistet, daher werden keine geheimen Schlüssel übertragen. In GSI-basierten Grid-Infrastrukturen wird eine Delegation in sieben Schritten wie folgt durchgeführt:

1. Gegenseitige Authentifizierung. Beide Partner authentifizieren sich mit ihren EEC oder Proxy-Zertifikaten beim jeweiligen Gegenüber.

2. Ankündigung des Delegationswunsches. Der Delegierende kündigt dem Delegationsempfänger an, dass er ihm Rechte delegieren will.
3. Generieren eines Schlüsselpaares. Der Delegationsempfänger generiert ein neues Schlüsselpaar, typischerweise mit Schlüsseln geringer Länge (512 bit).
4. Erstellen eines Zertifikatrequests. Der Delegationsempfänger erstellt einen Zertifikatrequest für ein Proxy-Zertifikat. Dazu verwendet er die Identitätsdaten des Delegierenden aus dem zur Authentifizierung verwendeten Zertifikat sowie den generierten öffentlichen Schlüssel. Der Zertifikatrequest wird mit dem neu generierten geheimen Schlüssel signiert.
5. Übertragung des Zertifikatrequests. Der Delegationsempfänger überträgt den Zertifikatrequest über den gesicherten Kanal.
6. Erstellen des Proxy-Zertifikats. Der Delegierende überprüft den erhaltenen Request auf Richtigkeit der enthaltenen Daten, erstellt ein neues Proxy-Zertifikat und signiert dieses mit seinem geheimen Schlüssel.
7. Übertragen des Proxy-Zertifikats. Das neue Proxy-Zertifikat wird zusammen mit allen übergeordneten (Proxy-)Zertifikaten zum Delegationempfänger zurückübertragen.

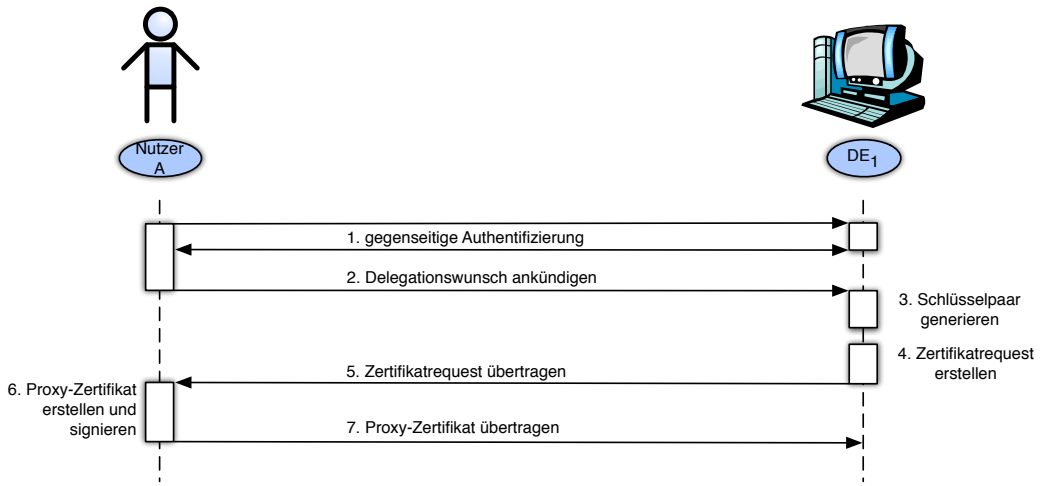


Abbildung 2.17: Delegation eines Proxy-Zertifikats

Zusammen mit dem zugehörigen geheimen Schlüssel kann sich der Delegationsempfänger ab diesem Zeitpunkt im Namen des Delegierenden authentifizieren und dessen Rechte ausüben.

2.3.4 Autorisierung auf Basis Virtueller Organisationen

In Kapitel 2.1.1 wurde die Virtuelle Organisation als Grid-spezifische Organisationsform für inter-institutionelle Arbeitsgruppen und Ressourcenverbünde vorgestellt. Virtuelle Organisationen bilden den organisatorischen und in zunehmendem Maße auch juristischen Rahmen, in dem Grid-Infrastrukturen betrieben und genutzt werden. Um der Nutzung von Ressourcen durch Virtuelle Organisationen eine rechtliche Basis zu geben, werden zwischen Ressourcenbetreibern und Virtuellen Organisationen *Service Level Agreements* (SLA) und *Acceptable Use Policies* (AUP) vereinbart. Erstere legen die zu erbringende Dienstgüte von Seiten der Ressourcenbetreiber gegenüber der VO fest. Die in den AUP festgelegten Richtlinien beschreiben, welche Aktivitäten auf den zur Verfügung stehenden Grid-Ressourcen ausgeführt werden dürfen und stellt somit eine rechtliche Absicherung der Ressourcenbetreiber dar. Um die rechtliche Bindung an ihre Mitglieder weiterzugeben, vereinbaren die Virtuellen Organisationen zusätzlich weitere AUP mit diesen.

Neben diesen Aspekten ist jedoch auch die technische Umsetzung des VO-Modells von Bedeutung. Dies ist zum Einen die Verwaltung von Nutzern, ihrer Rollen und Attribute innerhalb der Grid-Infrastruktur, zum Anderen die Zugriffskontrolle auf Basis der VO-Attribute auf Ressourcenebene.

2.3.4.1 Management Virtueller Organisationen als Attributautorität für Grid-Verbünde

Die Verwaltung von Nutzern innerhalb der VO wird durch spezialisierte Dienste vorgenommen. Diese dienen dem Verantwortlichen für die VO in erster Linie dazu, Arbeitsabläufe wie die Aufnahme neuer Mitglieder oder Änderungen des Status von Mitgliedern zu vereinfachen. In dynamischen Gruppen, in denen die Mitgliedschaften der Nutzer kurzlebig sind, ist ein solcher Dienst darüber hinaus unerlässlich, um eine konsistente Nutzerdatenbank zu erhalten.

Neben diesen grundlegenden Funktionalitäten, die für sich genommen noch keine besonderen Sicherheitsvorkehrungen erforderlich machen, werden VO-Managementdienste auch als Attributautorität (AA) eingesetzt. In diesem Fall verwaltet das VO-Management neben den persönlichen Daten eines Nutzers, wie seinem Namen, seiner Zugehörigkeit zu einer Institution oder seine Telefonnummer, auch zur Autorisierung von Zugriffen auf Grid-Ressourcen eingesetzte Informationen. Diese als VO-Attribute bezeichneten Informationen drücken Eigenschaften eines Nutzers, seine Position innerhalb der VO oder auch konkrete Privilegien zur Nutzung von Ressourcen aus.

Die heute eingesetzten Systeme zur Verwaltung von Virtuellen Organisationen stützen sich bei der Definition der Position des VO-Mitglieds in der Virtuellen Organisation in der Regel auf ein rollenbasiertes Modell (vgl. Kapitel 2.3.1.3). Die

VO-Attribute müssen folglich neben generischen Attributen auch die Rolle des Nutzers in einem hierarchischen VO-Modell wiedergeben können. In heutigen Grid-Infrastrukturen werden typischerweise drei Formate für Attribute in Systemen für das VO-Management verwendet:

1. **Attribut/Wert** Dieses Format bildet Paare aus einem eindeutigen Attributnamen und einem zugewiesenen Wert. Es kann verwendet werden, um eine Eigenschaft des Nutzers auszudrücken, z. B. seine Nationalität. Es lassen sich beliebig viele Attribut/Wert-Paare für einen Nutzer vergeben.
2. **FQAN** Der *Fully Qualified Attribute Name* (FQAN) kommt beim *Virtual Organisation Membership Service* (VOMS) [INF05] zum Einsatz. Er hat die Form

   $$< group > [/Role = [< role >][/Capability =< capability >]]$$

   und besteht aus der Angabe der Zugehörigkeit zu einer optionalen Untergruppe (*group*) innerhalb der VO. Weiterhin wird eine Rolle in der VO sowie zugeordnete Befähigungen (*capabilities*) spezifiziert. Sollen einem Nutzer keine über die Mitgliedschaft in der VO hinausgehenden Attribute zugeordnet werden, nehmen sowohl *Role* als auch *Capability* den Wert *Null* an. Der FQAN hat einen hierarchischen Aufbau, eine Rolle in einer Gruppe der VO kann somit andere Berechtigungen beinhalten als die gleiche Rolle in einer anderen Gruppe.
3. **Privileg** Privilegien ordnen einem Nutzer das Recht zu, eine Ressource auf eine definierte Zugriffsart zu nutzen. Dies kann z. B. der Lesezugriff auf eine bestimmte Datei innerhalb einer Speicherressource sein. Attribute dieser Form werden vom *Community Authorization Service* (CAS) [Glo06] verwendet.

Wird das VO-Management als Attributautorität verwendet, muss sichergestellt werden, dass die zur Autorisierung verwendeten Attribute von der Attributautorität ausgestellt und auf dem Weg zur Ressource nicht verändert wurden. Dies ist insbesondere dann der Fall, wenn die Übermittlung der Attribute zur Ressource per Push-Verfahren (vgl. Kapitel 2.3.2.2) durch den Nutzer erfolgt, also keine direkte, authentifizierte und gegen unerkannte Veränderung der transportierten Daten geschützte Verbindung zwischen Ressource und Attributautorität wie im Pull-Verfahren existiert. Um den Ursprung der Attribute überprüfen zu können, verwendet der VOMS signierte X.509-Attributzertifikate [FH02], die für den Transport zur Ressource in Proxy-Zertifikate eingebunden werden. Der *Community Authorization Service* nutzt SAML-Assertions [WAM+03], die ebenfalls digital signiert werden. Beide Formate sind auch gegen nicht zu erkennende Veränderung geschützt.

2.3.4.2 Zugriffskontrolle auf Ressourcenebene

Die letzte Entscheidung über die Gewährung oder Ablehnung des Zugriffs auf eine Grid-Ressource wird durch die Zugriffskontrolle der Grid-Ressource selbst getroffen.

Sie unterliegt in der Regel den Richtlinien der betreibenden Einrichtung, die sich durch Service Level Agreements zum Erbringen eines Dienstes in einer festgelegten Qualität verpflichtet hat. Als Basis für die Zugriffsentscheidung können Autorisierungsinformationen aus verschiedenen Quellen herangezogen werden. Diese Quellen werden durch die vier Hierarchieebenen im Grid differenziert:

Grid-Level Betreiber von Grid-Ressourcen sind in der Regel in einen Verbund integriert, der eine Grid-Infrastruktur für Nutzergruppen realisiert. Dieser Verbund verpflichtet sich, einen Dienst für eine oder mehrere Virtuelle Organisationen zu erbringen. Der einzelne Betreiber unterliegt den Richtlinien des Verbundes, die auf den Ressourcen in Form von Zugriffskontrollen realisiert werden müssen.

VO-Level Wie in Kapitel 2.3.4.1 erläutert, legen die Virtuellen Organisationen ihre Organisationsstruktur in Eigenverantwortung fest. Dem Nutzer werden signierte Bestätigungen über seine Attribute ausgestellt, die er den Ressourcen beim Zugriff auf diese vorlegt. Durch Verträge mit dem Betreiber der Grid-Ressource bzw. der Grid-Infrastruktur muss festgelegt werden, welche VO-Attribute zu welcher Dienstnutzung berechtigen. Diese Verträge werden in Richtlinien gefasst und müssen in der Zugriffskontrolle entsprechend abgebildet werden.

Site-Level Die Zugriffskontrolle auf Ebene des Ressourcenbetreibers setzt Richtlinien um, die für dessen gesamte Infrastruktur gelten. Diese können Regelungen für die Nutzung der Infrastruktur durch lokale Gruppen beinhalten, die keiner VO angehören. Weiterhin ist denkbar, dass in lokalen Richtlinien einzelne Nutzer ausgeschlossen werden, die Ressourcen in missbräuchlicher Weise verwendet haben.

Ressource-Level Der Ressource-Level entspricht dem Site-Level, schränkt aber die Reichweite der Richtlinie auf eine einzelne Ressource ein.

3 Analyse

Die Identifizierung von Schwachstellen bestehender Delegationsmechanismen in Grid-Umgebungen setzt die genaue Kenntnis der implementierten Authentifizierungs- und Autorisierungsmechanismen voraus. In diesem Kapitel wird daher eine detaillierte Betrachtung dieser Abläufe zur Laufzeit von Rechenaufträgen durchgeführt. Die Untersuchungen werden am Beispiel der gLite-Middleware durchgeführt, da diese eine Vielfalt an Diensten bereitstellt, die den Aufbau vollständiger Grid-Infrastrukturen aus Rechner- und Speicherressourcen ermöglichen. Aufgrund der Verteiltheit der zum Erbringen dieser Dienste benötigten Komponenten ist zudem zwingend der Einsatz von Delegationsmechanismen erforderlich. Die aus den Betrachtungen gewonnenen Erkenntnisse werden nachfolgend für eine Untersuchung und Bewertung bestehender Delegationsmechanismen und potentieller Schwachstellen der gLite-Middleware verwendet. Im dritten Abschnitt werden dann Kriterien aufgestellt, anhand derer diese Mechanismen mit Delegationsverfahren verglichen werden, die außerhalb der gLite-Middleware entwickelt werden bzw. bereits im Einsatz sind. Das Kapitel schließt mit einer Bewertung dieser Verfahren.

3.1 Autorisierungsvorgänge zur Laufzeit eines Rechenauftrags

Aus Sicht der Nutzer ist eine Hauptaufgabe von Grid-Infrastrukturen, ihre Rechenaufträge (*compute jobs*) effizient auf die vorhandenen Ressourcen zu verteilen und dort ausführen zu lassen. Die zu diesem Zweck implementierte, stark verteilte Architektur von *gLite* (vgl. Anhang A) erfordert den Einsatz fortgeschrittener Authentifizierungs- und Autorisierungsverfahren, sowohl bei der Interaktion von Grid-Diensten untereinander als auch bei der Interaktion von Nutzern mit Diensten. In diesem Kapitel werden die Authentifizierungs- und Autorisierungsvorgänge anhand eines exemplarischen Rechenauftrags analysiert.

3.1.1 Szenario eines Rechenauftrags

Das für die Analyse von Authentifizierungs- und Autorisierungsvorgängen definierte Szenario sieht vor, dass der Nutzer einen Rechenauftrag vom Typ „Normal“ erstellt. Dieser Job-Typ ermöglicht die Durchführung von Rechenaufträgen, die auf einem *Worker Node* ausgeführt werden und ist der einfachste in der gLite-Middleware vorgesehene. Die komplexeren Job-Typen unterscheiden sich von dem verwendeten in der Art, dass sie die Ausführung von Programmen auf mehreren

WN-Systemen ermöglichen. Dazu wird MPI zur Interprozesskommunikation verwendet. Die Abläufe der Authentifizierung und Autorisierung verändern sich jedoch nicht durch die Art des verwendeten Job-Typus, auch ist der Rechenauftrag dadurch nicht minder realitätsnah. Generell durchläuft ein Rechenauftrag in einer gLite-Infrastruktur drei Phasen:

1. Vorbereitung des Auftrags und Übergabe an die Grid-Infrastruktur bis zum Eintreffen auf dem WN.
2. Bearbeitung des Rechenauftrags auf dem WN.
3. Beendigung des Auftrags bis zum Abrufen der *Output-Sandbox*[1] durch den Nutzer.

Während der zweiten Phase, d. h. der Bearbeitung des Rechenauftrags auf dem WN, wird ein durch den Nutzer erstelltes Skript oder binäres Programm abgearbeitet. In diesem Szenario werden folgende Schritte durchgeführt:

1. Einlesen einer Eingabedatei, die auf einem SRM-System gespeichert ist. Der Zugriff auf den SRM-Rechner erfolgt über einen IO-Service, der für die Autorisierung von Anforderungen den *File Authorization Service* eines *FireMan Data Catalog* als PDP verwendet.
2. Durchführen der Berechnungen auf den Eingabedaten. Diese Berechnung dauert länger als zwölf Stunden. Dieser Umstand hat zur Folge, dass das zur Delegation der Nutzerprivilegien verwendete Proxy-Zertifikat erneuert werden muss. Dieser Vorgang, der als *Proxy-Renewal* bezeichnet wird, läuft ohne Nutzerinteraktion zwischen dem WMS und den Services MyProxy und VOMS ab.
3. Sichern der Ausgabedaten auf dem SRM-System nach dem gleichen Schema wie beim Einlesen der Eingabedatei.

3.1.2 Phase 1 – Bearbeitung des Auftrags durch die Grid-Middleware

In der ersten Phase des Rechenauftrags werden die Delegationen erstellt, die die Ausführung des Auftrags autorisieren. Der Auftrag wird in der *Job Description Language* (JDL) [Pac06] beschrieben und an die Grid-Umgebung übergeben. Im letzten Schritt dieser Phase wird eine passende Ressource gesucht und der Auftrag im Erfolgsfall an diese übergeben.

3.1.2.1 Initiierung der Delegationen

Bevor der Nutzer die notwendigen Delegationen erstellen kann, muss er zwei Voraussetzungen erfüllen, die nicht Bestandteil dieser Betrachtungen sind:

[1] Die *Output-Sandbox* umfasst im Vorhinein definierte Dateien, die nach Beendigung des Rechenauftrags durch die Grid-Infrastruktur zum Nutzer transportiert werden.

1. Besitz eines Nutzerzertifikates (EEC) von einer durch die VO anerkannten CA.
2. Mitgliedschaft in einer durch die Grid-Infrastruktur autorisierten VO.

Wie bereits in Kapitel 2.3.3.1 dargelegt, wird die Delegation von Nutzerprivilegien in Grid-Infrastrukturen technisch unter Verwendung von Proxy-Zertifikaten realisiert. Diese werden mit einer Gültigkeitsdauer von zwölf Stunden erstellt, so dass Rechenaufträge nach Ablauf dieser Frist über keine gültige Delegation verfügen und durch die Grid-Middleware beendet würden. In gLite-Infrastrukturen werden Rechenaufträge mit einer Laufzeit von mehr als zwölf Stunden durch den Proxy-Renewal-Mechanismus unterstützt, der in Kapitel 3.1.3.2 erläutert wird. Zu diesem Zweck führt der Nutzer U folgende Delegation aus:

$$keyEEC_U \quad \textbf{delegates} \quad mayRenew(PC_U,\, 168h,\, WMS)^{\wedge} * \quad \textbf{to} \quad myproxy \tag{3.1}$$

Mit dieser Delegation, die in *Delegation Logic* [Li00] formuliert ist, überträgt der Nutzer U der Entität *myproxy* das Privileg, für 168 Stunden (eine Woche) dem eindeutig identifizierten Dienst WMS eine neue Delegation auszustellen. Der Dienst WMS muss zusätzlich im Besitz eines gültigen Proxy-Zertifikats des Nutzers PC_u sein. Eine Beschränkung auf bestimmte Proxy-Zertifikate des Nutzers ist in Ermangelung von eindeutigen Merkmalen, wie einer Seriennummer, unmöglich. Das an ihn delegierte Privileg kann von *myproxy* an andere Entitäten redelegiert werden. Dies wird durch eine Delegationstiefe von $*$ ($^{\wedge}*$) ausgedrückt. Die Delegationstiefe kann durch den Nutzer in der aktuellen Implementierung nicht beschränkt werden.

Neben der Delegation an den MyProxy-Dienst für das Proxy-Renewal benötigt der Nutzer für einen Rechenauftrag ein Proxy-Zertifikat, in das ein durch den VOMS seiner VO ausgestelltes Attributzertifikat (AC) eingebettet ist. Dieses beglaubigt seine VO-Mitgliedschaft und dient auf den Grid-Ressourcen der VO- und rollenbasierten Zugriffskontrolle. Es ist durch den VOMS signiert und hat eine eigene Gültigkeitsdauer. Formal drückt Nutzer U, der ein Proxy-Zertifikat mit eingebetteten VOMS-Attributzertifikaten erstellt, folgende Delegation aus:

$$keyEEC_U \quad \textbf{delegates} \quad isMember((U\,\textbf{in}\,R)_{VO_x})^{\wedge} * \quad \textbf{to} \quad keyPC_U \tag{3.2}$$

Damit delegiert der Nutzer U das Privileg, in seinem Namen in der Rolle R im Kontext von VO_x zu handeln, an den geheimen Schlüssel des Proxy-Zertifikats PC_U. Auch diese Delegation erfolgt mit unbeschränktem Privileg zur Redelegation. Bei Zugriffen auf Grid-Ressourcen unter Verwendung dieses Proxy-Zertifikats hängt die interpretierte Bedeutung der Delegation von der verwendeten Autorisierungsmethode ab. Für Ressourcen, die VOMS-Attribute für die Zugriffskontrolle verwenden, bedeutet die Delegation:

$$keyPC_U \quad \textbf{speaks\_for} \quad (U\,\textbf{in}\,R)_{VO_x} \tag{3.3}$$

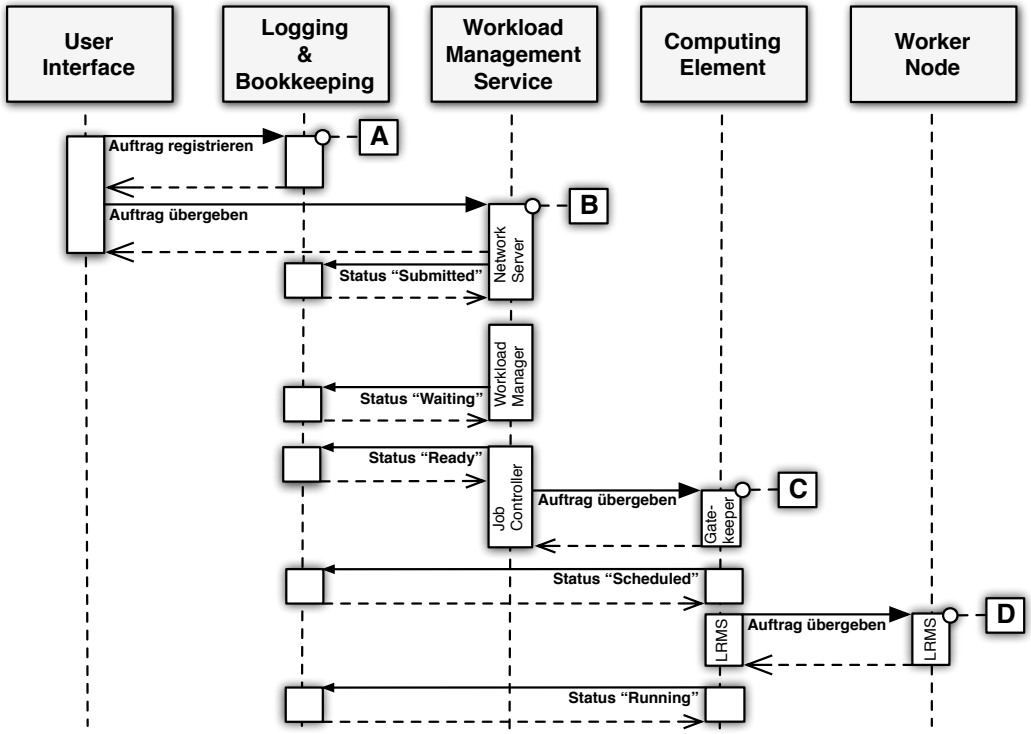


Abbildung 3.1: Ablauf eines Rechenauftrags – Phase 1

Ressourcen, deren Zugriffskontrolle keine VOMS-Attributzertifikate berücksichtigen, interpretieren die Delegation als:

$$keyPC_U \quad \textbf{speaks\_for} \quad U \tag{3.4}$$

Die erweiterte Funktionalität von Ressourcen, die VO-Attribute im Rahmen der Zugriffskontrolle interpretieren können, ermöglicht die Unterscheidung von multiplen Rollen R_x und damit eine Ableitung unterschiedlicher Privilegien für Mitglieder derselben VO. Ressourcen ohne diese Funktionalität können selbst bei Abgleich der Mitgliederlisten der VO mit den zur Zugriffskontrolle verwendeten Listen auf der Ressource nur eine einfache Zugriffsentscheidung nach Mitgliedschaft vornehmen. Eine Unterscheidung zwischen verschiedenen Aufgaben innerhalb der VO ist damit nicht möglich. Sollte der Nutzer Mitglied in mehreren auf der Ressource akzeptierten Virtuellen Organisationen sein, ist damit die Unterscheidung, in welcher dieser Organisationen er gerade aktiv ist, unmöglich.

3.1.2.2 Übergabe des Rechenauftrags an die Grid-Infrastruktur

Der in der JDL formulierte Rechenauftrag wird nachfolgend durch den Nutzer vom UI aus an das WMS übergeben. Dieser Vorgang läuft in zwei Schritten, wie in Abbildung 3.1 auf der vorherigen Seite dargestellt, ab:

1. **Schritt A** Registrierung des Auftrags mit einer eindeutigen *JobID* bei einem LB-Service. Dieser autorisiert den Nutzer anhand des *Distinguished Names* im Subjekt des zur Authentifizierung verwendeten Proxy-Zertifikats. Der DN des Nutzers muss in der zur Autorisierung herangezogenen Datei, dem *grid-mapfile*, enthalten sein. Das *grid-mapfile* wird regelmäßig mit den Nutzerlisten der unterstützten VOs und deren VOMS-Servern abgeglichen. Die Interpretation der Delegation erfolgt folglich wie in Formel 3.4 dargestellt.
2. **Schritt B** Übergabe des Auftrags an den Network-Server des WMS und Ableitung eines weiteren Proxy-Zertifikats nach dem in Kapitel 2.3.3.1 dargelegten Verfahren. Anschließend werden Dateien, die der Nutzer für die Bearbeitung seines Auftrags auf dem WN benötigt, mittels *GridFTP* auf das WMS übertragen. Für beide Vorgänge laufen Autorisierungsentscheidungen nach dem in Schritt A beschriebenen Muster ab.

3.1.2.3 Weiterleiten des Rechenauftrags an ein Computing Element

Nach der Übergabe des Auftrags an den Network-Server des WMS wird durch den *Workload Manager* eine passende CE-Ressource ermittelt. Dies geschieht durch Abgleich der in der JDL-Datei spezifizierten Anforderungen mit den Daten aller am WMS angemeldeten CE-Ressourcen. Ist ein passendes CE gefunden, wird der Auftrag an den Job-Controller weitergeleitet, der den Auftrag in zwei Schritten an das CE übergibt:

1. **Schritt C** Authentifizierung und Autorisierung am *Gatekeeper* des CE unter Verwendung des für das WMS für diesen Auftrag abgeleiteten Proxy-Zertifikates (Schritt C in Abbildung 3.1 auf der vorherigen Seite). Die Autorisierung erfolgt im LCAS-Modul unter Berücksichtigung der VO-Attribute. Die Interpretation der Delegation erfolgt wie in Formel 3.3 dargestellt. Auf dem CE wird ein weiteres Proxy-Zertifikat von dem an das WMS delegierten abgeleitet.
2. Nach erfolgreicher Autorisierung wird auf dem CE ein Condor-Prozess [Con07] gestartet, der den Auftrag von einer Condor-Instanz auf dem WMS bezieht (nicht in Abbildung 3.1 dargestellt).

Das CE kontrolliert über ein *Local Resource Management System* (LRMS) die angeschlossenen WN-Systeme. Das LRMS, im Falle der gLite-Testumgebung am RRZN ein Batch-System auf Basis der TORQUE-Software [Clu07], ist kein Bestandteil

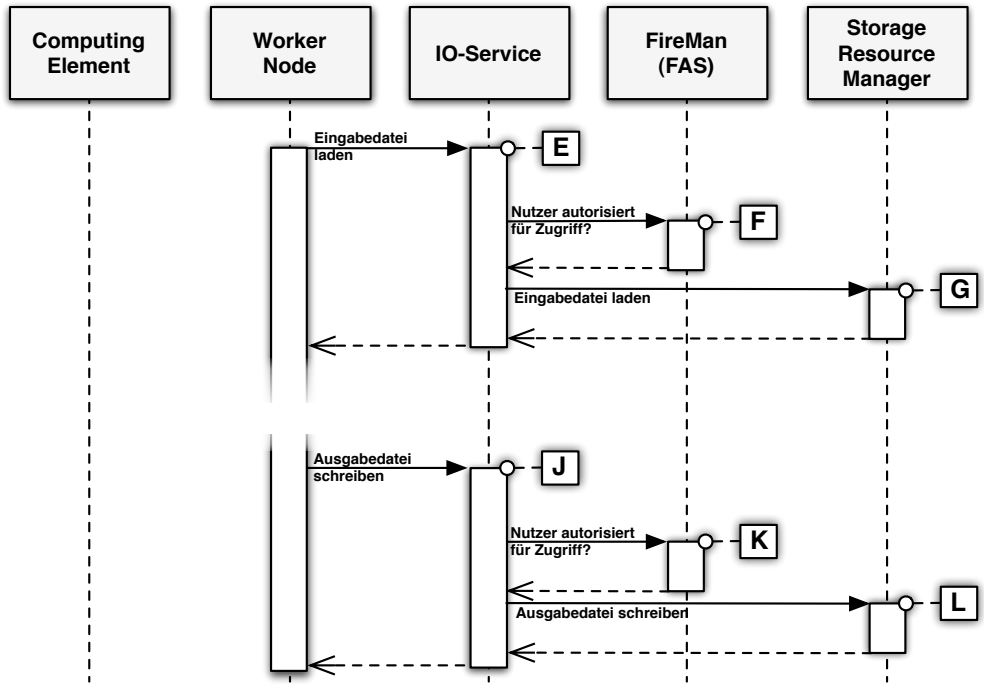


Abbildung 3.2: Ablauf eines Rechenauftrags – Phase 2

der gLite-Middleware. Für die Weiterleitung des Rechenauftrags an die *Worker Nodes* (Schritt D von Abbildung 3.1) werden keine Authentifizierungs- und Autorisierungsvorgänge mit Mechanismen der Grid-Middleware durchgeführt. Die vom Batch-System verwendeten Verfahren basieren auf Public-Key-Mechanismen der *Secure Shell*. Um den Rechenauftrag jedoch in die Lage zu versetzen, Operationen auf Grid-Ressourcen für den Nutzer durchführen zu können, wird ein Proxy-Zertifikat von dem an das CE delegierte abgeleitet und mit dem Rechenauftrag zu dem verwendeten WN übertragen.

3.1.3 Phase 2 – Bearbeitung des Rechenauftrags auf dem Worker Node

Die Bearbeitung des Rechenauftrags findet auf dem WN unter Kontrolle des LRMS statt. Die Arbeitsumgebung ist ein UNIX-Account, der dem Nutzer für die Dauer der Bearbeitung zur Verfügung steht. Es können Programme verwendet werden, die für den UNIX-Account zugreifbar sind, zudem steht es dem Nutzer offen, eigene Programme auf dem System zu übersetzen oder als Kompilat in der *Input-Sandbox* zu übertragen. Die für den Auftrag benötigten Eingabedaten werden im Regelfall von externen Speichersystemen bezogen.

3.1.3.1 Kommunikation mit den Datendiensten

Die betrachteten Datendienste der gLite-Middleware bestehen aus den drei in Kapitel 2.1.3.2 vorgestellten Komponenten. Für die Übertragung von Dateien von und zu einem *Storage Element* (SE) mit SRM-Schnittstelle kommunizieren die Programme des gLite-Datenmanagement ausschließlich mit dem IO-Service. Für die Übertragung einer Datei sind jeweils drei Authentifizierungs- und Autorisierungsvorgänge (vgl. Abbildung 3.2 auf der vorherigen Seite) erforderlich:

1. **Schritt E, J** Für den Zugriff auf den IO-Service ist eine Authentifizierung und Autorisierung unter Verwendung des Proxy-Zertifikats notwendig, dass zusammen mit dem Rechenauftrag vom CE zum WN übertragen wurde. Die Autorisierung erfolgt identitätsbasiert, wie in Schritt A in Kapitel 3.1.2.2 dargelegt. Es wird eine weitere Delegation durchgeführt. Mit diesem Proxy-Zertifikat finden die in Schritt F und K dargestellten Zugriffe auf den FireMan Data Catalog statt. Die Interaktion mit dem IO-Server ist erst nach der vollständigen Übertragung der Datei abgeschlossen.
2. **Schritt F, K** Die Autorisierung des Nutzers zum Zugriff auf die Datei wird anhand der Dateirechte sowie zusätzlicher ACLs des *File Authorization Service* (FAS) überprüft. Den Objekten des *File Catalog* Dienstes sind individuelle Zugriffsprivilegien für den Nutzer, seine Gruppe sowie Dritte zugeordnet. Die Gruppe bilden jeweils alle Nutzer, deren VO-Attribute identisch sind. Die Zugriffsprivilegien sowie die optionale ACL für weitere Entitäten werden durch den Besitzer des Objektes konfiguriert. In Schritt K wird zusätzlich ein neuer Eintrag für die Zieldatei im *File Catalog* angelegt.
3. **Schritt G, L** Hat der Nutzer die erforderliche Berechtigung, greift der IO-Service unter Verwendung seines eigenen X.509-Zertifikates auf den SRM-Service zu. Die Autorisierung auf diesem erfolgt identitätsbasiert und gestattet lediglich dem IO-Service den Zugriff.

3.1.3.2 Erneuerung der Proxy-Credentials

Wie in Kapitel 3.1.2.1 dargelegt, unterstützt *gLite* das automatische Erneuern von Proxy-Zertifikaten. Die Notwendigkeit für das Proxy-Renewal entsteht bei Bearbeitungsdauern eines Rechenauftrags, die die Gültigkeitsdauer des delegierten Proxy-Zertifikates überschreiten. Der Ablauf des Proxy-Renewals ist wie folgt:

1. Bei Übergabe des Auftrags an das WMS wird der Auftrag am Proxy-Renewal-Service registriert.
2. **Schritt H** Wird die Restgültigkeitsdauer des delegierten Proxy-Zertifikates von 20 Minuten unterschritten, kontaktiert der Proxy-Renewal-Service den MyProxy-Service und authentifiziert sich dort mit seinem EEC und dem

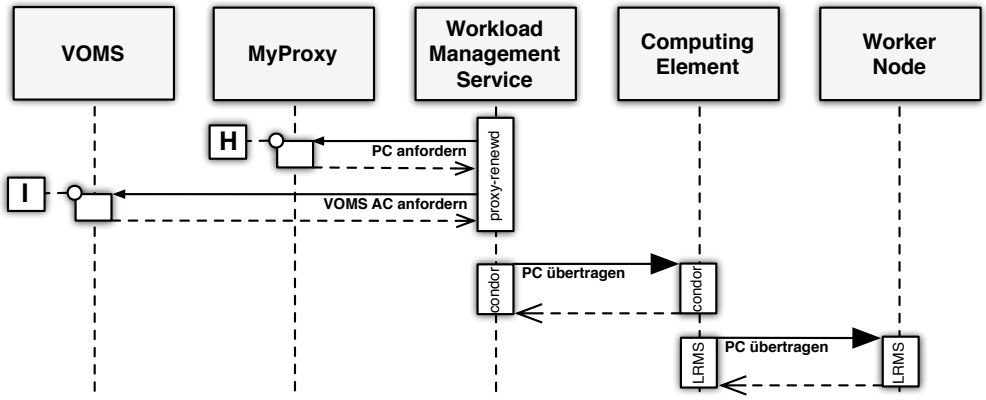


Abbildung 3.3: Ablauf des Proxy-Renewal in gLite-Infrastrukturen

Proxy-Zertifikat des Nutzers. Ist der DN des WMS in der dem Proxy-Zertifikat zugehörigen ACL (vgl. Formel 3.1) enthalten, leitet der MyProxy-Service ein neues Proxy-Zertifikat ab und überträgt es zum WMS.

3. **Schritt I** Das delegierte Proxy-Zertifikat enthält keine VO-Attribute. Daher bezieht der Proxy-Renewal-Service nachfolgend unter Authentifizierung mit dem neuen Proxy-Zertifikat vom VOMS-Server ein neues Attributzertifikat. Die beantragten VO-Attribute entsprechen denen des ursprünglichen Proxy-Zertifikates. Hat sich die VO-Mitgliedschaft des Nutzers in der Zwischenzeit nicht geändert, erstellt der VOMS das neue Attributzertifikat. Nachfolgend leitet der Proxy-Renewal-Service ein weiteres Proxy-Zertifikat von dem erhaltenen unter Einbindung des neuen AC ab.
4. Das Proxy-Zertifikat wird nun verwendet, um eine neue Delegation für das CE abzuleiten und dorthin zu übertragen. Dieser Vorgang wird für den WN wiederholt.

3.1.4 Phase 3 – Abschluss des Rechenauftrags

Nach Abschluss der Bearbeitung des Rechenauftrags auf dem WN werden die vom Nutzer spezifizierten Dateien in der *Output-Sandbox* auf das WMS transferiert. Aus Redundanzgründen werden die Dateien sowohl über das LRMS und damit das CE (Schritt M in Abbildung 3.4 auf der nächsten Seite) als auch per gridftp direkt zum WMS (Schritt N) übertragen. Nach Erhalt der Dateien setzt das WMS den Status „Done“ für diesen Auftrag auf dem LB-Server. Nun kann der Nutzer diese Dateien mittels gridftp vom WMS kopieren und damit den Auftrag abschließen. Die in den Schritten M und N angewendeten Autorisierungsverfahren wurden in Kapitel 3.1.2.2 und 3.1.2.3 dargelegt.

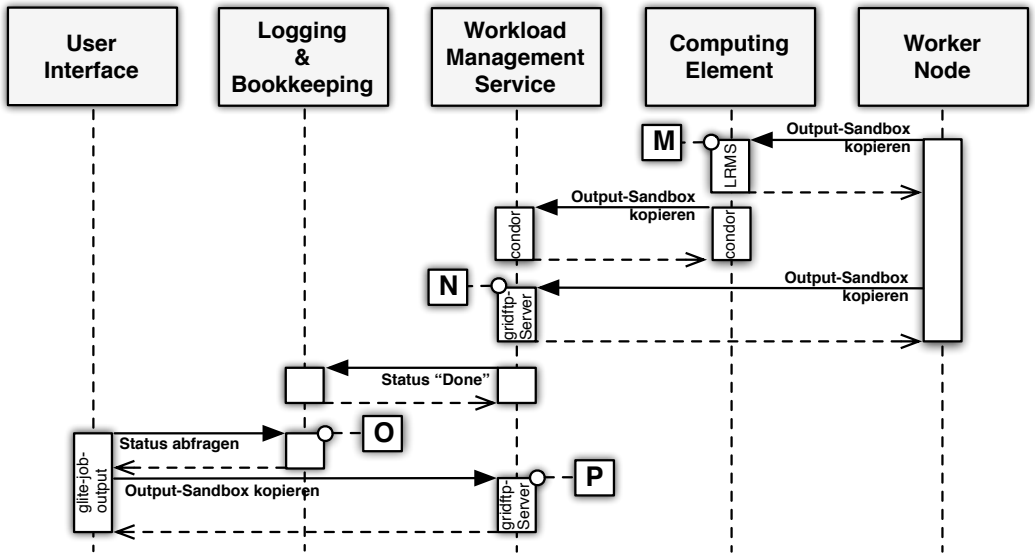


Abbildung 3.4: Ablauf eines Rechenauftrags – Phase 3

3.2 Delegation von Privilegien in gLite-Infrastrukturen

In diesem Kapitel wird eine Bewertung der Delegation von Nutzerprivilegien in gLite-Umgebungen vorgenommen. Diese erfolgt auf Basis der in Kapitel 3.1 untersuchten Authentifizierungs- und Autorisierungsabläufe. Neben allgemeinen Aspekten der Verwendung von Proxy-Zertifikaten als Medium für Delegationen werden die Dienste der gLite-Middleware im Einzelnen betrachtet. Nachfolgend werden Szenarien diskutiert, die zu einer Kompromittierung der delegierten Nutzerprivilegien führen können. Dieser Abschnitt schließt mit einer Bewertung der implementierten Delegationsmechanismen.

3.2.1 Einsatz von Proxy-Zertifikaten

Grid-Infrastrukturen, deren Authentifizierungs- und Autorisierungsinfrastruktur auf der GSI (vgl. Kapitel 2.2.3) basiert, verwenden Proxy-Zertifikate zur Delegation von Nutzerprivilegien. Einer Reihe von Vorteilen dieses Ansatzes stehen jedoch auch gravierende Nachteile gegenüber. Die inhärenten Vorteile des Einsatzes von Proxy-Zertifikaten für *Single Sign-On* und die Delegation von Nutzerprivilegien sind in folgenden Eigenschaften begründet:

- **Eigenes Schlüsselpaar** Jedes Proxy-Credential[2] verfügt über ein eigenes

[2] Als Proxy-Credential wird die Kombination aus Proxy-Zertifikat und zugehörigem geheimen Schlüssel sowie den ausstellenden Zertifikaten exklusive CA-Zertifikat bezeichnet.

Schlüsselpaar. Eine unerwünschte Offenlegung des geheimen Schlüssels verursacht keine Kompromittierung weiterer Zertifikate.

- **Flexible Gültigkeitsdauer** Proxy-Zertifikate verfügen über eine eigene, vom ausstellenden Zertifikat nur in Form einer oberen Schranke abhängigen Gültigkeitsdauer. Dadurch können Proxy-Zertifikate für zeitlich begrenzte Delegationen mit einem entsprechend kurzen Gültigkeitszeitraum versehen werden. In einigen Anwendungsfeldern wird so der durch eine Kompromittierung entstehende Schaden limitiert. Dies gilt aus offensichtlichen Gründen nicht für die Einsichtnahme in vertrauliche Daten Dritter.
- **Verifizierbare Vertrauenskette** Mit der Vertrauenskette (*chain of trust*) wird die verifizierbare Abfolge von Signaturen bezeichnet. Ein Zertifikat wird nur dann erfolgreich verifiziert, wenn alle Signaturen der ausstellenden Zertifikate bis zu der Zertifizierungsstelle überprüft werden können. Im Bereich der Proxy-Zertifikate liegt der zusätzliche Vorteil darin begründet, dass die in ein Zertifikat eingebetteten Attribute nicht durch Entfernen desselben aus der Kette unterdrückt werden können. Ein solcher Vorgang unterbricht die Vertrauenskette und verhindert für alle Zertifikate unterhalb des entfernten eine erfolgreiche Verifizierung.

Aus der Implementierung von Proxy-Zertifikaten ergeben sich unmittelbar eine Reihe von Nachteilen:

- **Keine eindeutige Adressierbarkeit** Proxy-Zertifikate werden ohne die bei Zertifizierungsstellen üblichen Seriennummern ausgestellt. Das Fehlen einer eindeutigen Adressierbarkeit sorgt dafür, dass der Nutzer einen größeren Umfang an Privilegien delegieren muss als notwendig. Ein Beispiel für eine solche Delegation ist die in Kapitel 3.1.2.1 erläuterte Delegation des Privilegs zur Erneuerung von Proxy-Zertifikaten an einen MyProxy-Dienst. Durch eine eindeutige Seriennummer im Kontext des ausstellenden Nutzers wäre eine gezielte Delegation dieses Privilegs möglich.
- **Kein Widerruf möglich** Zertifizierungsstellen erstellen in regelmäßigen Abständen signierte Sperrlisten für Zertifikate (vgl. Kapitel 2.2.4). Der Widerruf von Proxy-Zertifikaten ist auf diesem Wege nicht möglich, da sie nicht von Zertifizierungsstellen ausgestellt werden. Theoretisch wäre ein Widerruf möglich, den jeder Nutzer für die von ihm ausgestellten Proxy-Zertifikate durchführt. Für einen derartigen Mechanismus ist jedoch ein eindeutiges Merkmal, wie eine Seriennummer, in den Proxy-Zertifikaten vorzusehen.
- **Kein Schutz des geheimen Schlüssels** Das *Single Sign-On* und die Delegation von Nutzerprivilegien auf Basis von Proxy-Zertifikaten funktionieren nur dann, wenn der zugehörige geheime Schlüssel unverschlüsselt auf der Ressource vorliegt. Unter dieser Prämisse ist das Proxy-Credential jedoch ohne Mitwirkung des ausstellenden Nutzers im Klartext kopierbar und kann bis zum Ablauf der Gültigkeit uneingeschränkt verwendet werden.

Die dargestellten Eigenschaften betreffen die Nutzung von Proxy-Zertifikaten im Allgemeinen, sind ergo unabhängig von der verwendeten Grid-Middleware. Der größte konzeptionelle Mangel von Proxy-Zertifikaten besteht in der Kombination aus fehlender Widerrufsmöglichkeit und mangelhaften Schutzmechanismen des geheimen Schlüssels. Der fehlende Schutz des geheimen Schlüssels ermöglicht einen Missbrauch bei Zugriff auf die Datei, in der das Proxy-Credential gespeichert ist. Ohne die Möglichkeit zum Widerruf von Proxy-Zertifikaten kann ein Missbrauch – selbst wenn er entdeckt wurde – nicht oder nur unter Anwendung radikaler Maßnahmen, wie dem Widerruf des Nutzerzertifikates oder dem Ausschluss des Nutzers von der Nutzung der Grid-Ressourcen, begrenzt werden. Lediglich aus der relativ beschränkten Gültigkeitsdauer ergibt sich ein gewisser Schutz gegen den Missbrauch der Privilegien und die unerwünschte Einsichtnahme in die Daten des Nutzers.

Spezifisch für die gLite-Middleware ist die Art des implementierten Delegationskonzeptes. Die Delegation erfolgt implizit, d. h. der Nutzer kann aus seinen VO- und Rollen-Mitgliedschaften diejenigen auswählen, die er als signierte VO-Attribute in das Proxy-Zertifikat einbetten will. Die Interpretation dieser Delegation erfolgt lokal auf den Ressourcen und hängt von den dort implementierten Zugriffskontrollmechanismen ab (vgl. Kapitel 3.1.2.1). Die gLite-Middleware gibt dem Nutzer keine Möglichkeit, explizit zu formulieren, welche Privilegien er zu delegieren wünscht. Eine Beschränkung der Delegation auf spezifizierte Ressourcen ist ebenso nicht möglich. Aus diesen Beschränkungen ergibt sich in der Regel ein Übermaß an delegierten Privilegien, wodurch das Prinzip der minimalen Rechtevergabe [SS75] verletzt wird.

3.2.2 Mechanismen zur Sicherung delegierter Proxy-Credentials

In Kapitel 3.2.1 wurde dargelegt, dass Proxy-Credentials unverschlüsselt im Dateisystem der jeweiligen Grid-Ressource vorliegen, um von dieser ohne Nutzerinteraktion verwendbar zu sein. Der Schutz dieser Dateien durch geeignete Zugriffskontrollverfahren ist somit gleichbedeutend mit dem Schutz der delegierten Privilegien vor missbräuchlicher Nutzung durch Dritte. Da die Zugriffe auf das Dateisystem durch das Betriebssystem kontrolliert werden, ist eine Analyse dieses Umfeldes erforderlich. Eine Systemkonfiguration, die nach der guten fachlichen Praxis vorgenommen wurde, wird dabei vorausgesetzt. Dies schließt beispielsweise die Löschung nicht verwendeter Accounts, die Verwendung sicherer Passwörter, das Abschalten nichtgenutzter Dienste sowie das zeitnahe Einspielen sicherheitsrelevanter Aktualisierungen der eingesetzten Software ein.

Für die Sicherheitsanalyse der delegierten Proxy-Credentials müssen Fragen zur Konfiguration von Schutzmechanismen sowohl auf der Ressource als auch im Netz der Einrichtung gestellt werden. Die Antworten auf diese Fragen werden in Kapitel 3.2.3 für die Diskussion von Angriffsvektoren verwendet. Für jede Komponen-

te in einer gLite-Umgebung, die Delegationen von Nutzerprivilegien in Form von Proxy-Credentials erhält, müssen folgende Fragen gestellt werden:

- Welchem Account und welcher UNIX-Gruppe werden die Proxy-Credentials auf der jeweiligen Ressource zugeordnet?
- Welche Zugriffsrechte werden für die Proxy-Credentials konfiguriert?
- Werden auf der Ressource Pool-Accounts[3] eingesetzt?
- Haben die Grid-Nutzer Zugriff auf die UNIX-Ebene der Ressource?
- Bietet die Ressource Dienste an, die von externen Netzen aus zugreifbar sein müssen oder muss die Ressource auf externe Grid-Dienste zugreifen?

Mit der Klärung dieser Fragen kann das Risiko einer Offenlegung von kritischen Authentifizierungs- und Autorisierungsinformationen an unbefugte Dritte eingeschätzt werden. Eine Auswertung für die relevanten Dienste ist in Tabelle 3.1 dargestellt, die in drei Abschnitte unterteilt ist. Der erste Abschnitt führt die Art des Accounts auf, der für Grid-Nutzer verwendet wird. Weiterhin kann ihm entnommen werden, ob Nutzer über direkten Zugriff auf eine UNIX-Shell verfügen. Der zweite Abschnitt enthält die UNIX-Rechte der delegierten Proxy-Credentials sowie den Besitzer und die Gruppenzuordnung der Datei. Pool-Accounts sind mit *PA*, Pool-Groups mit *PG* gekennzeichnet. Im letzten Abschnitt wird die Notwendigkeit von Kommunikation mit externen Diensten im Falle eines institutionsübergreifenden Grid-Verbundes beantwortet. Es wird davon ausgegangen, dass die an der Grid-Infrastruktur beteiligten Institutionen mindestens über eigene UI- und MyProxy-Komponenten verfügen, die zentralen Komponenten jedoch in der Regel außerhalb der Institution des Nutzers lokalisiert sind.

3.2.3 Angriffsvektoren zur Offenlegung von Proxy-Credentials

Die in Tabelle 3.1 dargestellten Eigenschaften der Grid-Komponenten ermöglichen die Entwicklung von Angriffsszenarien, die das Ziel verfolgen, Proxy-Credentials Dritter zu erlangen. Angriffsvektoren können prinzipiell nach folgenden Eigenschaften unterschieden werden:

- **Lokaler oder entfernter Angriff** Der lokale Angriff setzt voraus, dass der Angreifer bereits über eine UNIX-Shell auf der Ziel-Ressource verfügt. Er muss folglich entweder im Vorfeld einen erfolgreichen entfernten Angriff ausgeführt haben oder für die Nutzung der Ressource u.U. mit geringeren Privilegien als angestrebt berechtigt sein. Es wird keine Aussage darüber getroffen, ob der Angreifer über Administrator-, so genannte Root-Privilegien, verfügt. Der entfernte Angriff wird auf Netzdienste des Betriebssystems oder

[3] Als Pool-Accounts werden Nutzerkonten bezeichnet, die von mehreren Nutzern einer Ressource, jedoch nicht zeitgleich verwendet werden.

	UI	WMS	CE	WN	IO-Service	MyProxy
verwendeter Account-Typ						
Nutzer-Account	ja	nein	nein	nein	nein	nein
Pool-Account	nein	ja	ja	ja	nein	nein
Zugriff auf UNIX-Shell	ja	nein	nein	ja	nein	nein
UNIX-Rechte der Proxy-Credentials						
Besitzer	Nutzer (r w -)	Dienst (r w -)	PA (r w -)	PA (r w -)	Dienst (r w -)	root (r w -)
Gruppe	Nutzer (- - -)	Dienst (- - -)	PG (- - -)	PG (- - -)	Dienst (- - -)	root (- - -)
andere	(- - -)	(- - -)	(- - -)	(- - -)	(- - -)	(- - -)
Kommunikation mit externen Ressourcen						
Angebot von Grid-Diensten	nein	ja	ja	nein	ja	ja
Zugriff auf Grid-Dienste	ja	ja	ja	ja	ja	nein

Tabelle 3.1: Eigenschaften von gLite-Komponenten in Bezug auf die Sicherheit von Delegationen

der Anwendungsschicht mit dem Ziel geführt, eine lokale UNIX-Shell zu erhalten. Angriffe mit dem Ziel der Dienstverweigerung (*denial of service*) werden in dieser Arbeit nicht betrachtet, da sie nicht zur Offenlegung von Proxy-Credentials genutzt werden können.

- **Grid-spezifischer oder allgemeiner Angriff** Der Grid-spezifische Angriff erfolgt mit Programmen oder auf Dienste der Grid-Middleware, der allgemeine Angriff setzt kein Vorhandensein einer Grid-Umgebung vorraus.

Aus diesen Kriterien lassen sich vier Kombinationen für Angriffsvarianten ableiten, deren Anwendbarkeit auf die verschiedenen Grid-Komponenten zu untersuchen ist. Die Abschätzung des Risikos für die Komponenten erfolgt dabei anhand der in Tabelle 3.1 aufgeführten Kriterien. Auf eine tabellarische Auflistung konkreter Angriffsvektoren wird verzichtet, da diese in der Regel stark von den verwendeten Versionen sowohl des Betriebssystems als auch der Anwendungssoftware abhängen und somit keine allgemeingültige Aussage getroffen werden kann.

Allgemeine, entfernte Angriffe Entfernte Angriffe zielen auf Schwachstellen der Implementierung des attackierten Dienstes, wie Pufferüberläufe (*buffer overflow*, vgl. [Kle03]) mit dem Ziel, sich lokalen Zugriff auf das System zu verschaffen. Angriffe auf Schwachstellen in den Implementierungen sind heute weit verbreitet und stellen eine reales Risiko für den hochverfügbaren Betrieb eines Dienstes und die

Sicherheit der auf diesem abgelegten Daten dar. Grid-Ressourcen sind von dieser Art des Angriffs nur dann betroffen, wenn sie neben ihren Grid-spezifischen auch allgemeine Dienste zur Verfügung stellen. Darunter fallen Dienste zur entfernten Administration wie *Secure Shell* oder auch Informationsdienste wie Web-Server. Das Risiko einer Kompromittierung kann bei der Vielzahl denkbarer Dienste nicht allgemeingültig bewertet werden. Auf den Einsatz nicht Grid-bezogener Dienste auf Grid-Ressourcen, die unverschlüsselte Proxy-Credentials speichern, sollte nach Möglichkeit verzichtet werden. Wenigstens sollte der Zugriff auf diese jedoch durch geeignete Maßnahmen beschränkt werden.

Grid-spezifische, entfernte Angriffe Diese Art des Angriffs kann prinzipiell auf vier der in Tabelle 3.1 aufgeführten gLite-Komponenten durchgeführt werden, UI und WN stellen keine Grid-Dienste bereit. Der MyProxy-Service stellt durch die Speicherung von Proxy-Credentials einer Vielzahl von Grid-Nutzern ein besonders wertvolles Ziel dar. Auch bekommt der Angreifer durch das Ausnutzen einer Sicherheitslücke im MyProxy-Dienst Administratorrechte, da dieser mit den Privilegien des Nutzers „root“ betrieben wird. Der Zugriff kann jedoch wirksam mit einer Firewall beschränkt werden, da auf ihn von außerhalb der betreibenden Institution nur durch das WMS zugegriffen werden muss (vgl. Kapitel 3.1.3.2). Für die Komponenten WMS, CE und IO-Service ist eine wirksame Beschränkung sowohl auf IP-Adressbereiche als auch auf Dienste kaum möglich. Zum Einen muss der Zugriff auf diese von allen an der Grid-Infrastruktur beteiligten Institutionen möglich sein, zum Anderen bieten diese Komponenten eine Vielzahl von Diensten an (vgl. Abbildung A.1 auf Seite 130), für die auch weite TCP-Portbereiche zugänglich sein müssen. Somit ist davon auszugehen, dass Sicherheitslücken in diesen Diensten der Grid-Middleware ein ernsthaftes Risiko darstellen. Die Existenz von Sicherheitslücken in gLite-Diensten wurde nachgewiesen [Gri08], bei komplexen Software-Projekten wie der gLite-Middleware sind weitere statistisch wahrscheinlich.

Allgemeine, lokale Angriffe Der allgemeine, lokale Angriff kann auf allen Grid-Ressourcen ausgeführt werden, die entweder durch einen entfernten Angriff kompromittiert sind oder durch legitime Nutzer, die sich erweiterte Privilegien verschaffen wollen (*privilege escalation attack*). Der triviale, jedoch in seinen Auswirkungen gravierende Fall, dass der Administrator eines Systems Proxy-Credentials eines Grid-Nutzers missbraucht, wird nicht betrachtet. Diese Missbrauchsvariante kann lediglich durch juristische Vorkehrungen begrenzt werden. Der lokale Angriff ohne vorhergehenden entfernten Angriff setzt den Zugang zu einer UNIX-Shell voraus. Dies ist lediglich auf den Komponenten UI und WN gegeben. Das auf den beiden Systemen eingesetzte Betriebssystem *Redhat Enterprise Linux Advanced Server 3* kann generell als sicher eingestuft werden. Der Hersteller reagiert zeitnah auf Sicherheitslücken und gibt entsprechende *Security Advisories* [RH07] heraus. Das

	UI	WMS	CE	WN	IO-Service	MyProxy
Entfernte Angriffe						
allgemein	+	+	+	+	+	+
Grid-spezifisch	−	+	+	−	+	++
Lokale Angriffe						
allgemein	+	−	−	+	−	−
Grid-spezifisch	−	−	−	++	−	−

Tabelle 3.2: Risikobewertung von gLite-Komponenten in Bezug auf die Sicherheit von Delegationen

Ausnutzen einer unentdeckten Sicherheitslücke kann jedoch nicht ausgeschlossen werden.

Grid-spezifische, lokale Angriffe Der Grid-spezifische, lokale Angriff macht sich Schwachstellen in der Architektur oder der Implementierung von Grid-Diensten zunutze. Es gelten die Aussagen zum allgemeinen lokalen Angriff. Für das *gLite*-UI sind keine Grid-spezifischen Angriffe bekannt. Im Falle der Worker Node Komponente kann ein Angreifer jedoch architekturspezifische Schwachstellen nutzen. Diese liegen in der Implementierung der Arbeitsumgebung von Grid-Nutzern mittels Pool-Accounts begründet. Auch nach Abschluss seines Rechenauftrags kann der Angreifer Prozesse mit den Privilegien des Pool-Accounts betreiben[4]. Diese können auf die Proxy-Credentials der nachfolgenden Nutzer zugreifen und dem Angreifer zugänglich machen. Die Gefahr der Ausnutzung solcher Grid-spezifischer Angriffe ist im Fall der Worker Node Komponente als real und gravierend zu bewerten.

Eine Übersicht über die Bewertung der gLite-Dienste, die Delegationen in Form von Proxy-Credentials empfangen, ist in Tabelle 3.2 aufgeführt. Vernachlässigbare Risiken oder solche, die von prinzipiell als unmöglich eingestuften Angriffsvarianten verursacht werden, sind mit „−“ gekennzeichnet. Den Diensten inhärente Risiken oder solche, die im konkreten Fall nicht nachgewiesen, aber statistisch wahrscheinlich sind, sind mit „+“ markiert. Risiken, die durch den verursachten Schaden besonders groß sind, wie im Falle einer Kompromittierung von MyProxy, werden durch „++“ gekennzeichnet. Dies gilt auch für Angriffe, die besonders einfach durchzuführen sind.

[4] Dies kann man z. B. durch Starten von Prozessen im Hintergrund mit dem Befehl `nohup` oder durch Eintrag eines *Cron-Jobs* erreichen.

3.2.4 Bewertung

Die Untersuchung der Delegation auf Basis von Proxy-Zertifikaten sowie ihrer Implementierung in den Diensten der gLite-Middleware in den Kapiteln 3.2.1 bis 3.2.3 hat erhebliches Missbrauchspotential aufgezeigt. Bedingt durch die Rolle des Nutzers als Aussteller und das Fehlen von Seriennummern sind Proxy-Zertifikate nicht widerrufbar. Die kurze Gültigkeitsdauer stellt lediglich einen geringen Schutz vor Missbrauch dar. Durch die Speicherung der Proxy-Credentials im Klartext werden die Ressourcen zu einem attraktiven Angriffsziel. Vor unberechtigtem Zugriff sollen hier die Zugriffskontrollmechanismen des Filesystems sorgen. Bei näherer Betrachtung erwiesen sich diese jedoch für einen wirksamen Schutz als unzureichend. Angriffsszenarien, die eine Kompromittierung von Proxy-Credentials ermöglichen, sind als praktisch durchführbar einzustufen.

Durch die Kompromittierung eines Proxy-Credentials erhält der Angreifer Zugriff auf die dem Nutzer zur Verfügung stehenden Grid-Ressourcen. So können Dateien des Nutzers auf einem SE gelesen oder gelöscht sowie weitere Rechenaufträge im Namen des Nutzers durchgeführt werden. Durch den Einsatz des automatischen Proxy-Renewal-Mechanismus wird das kompromittierte Proxy-Credential zudem ohne Benachrichtigung des Nutzers erneuert. Eine einmalige Kompromittierung ist somit bei Vorliegen einer Delegation auf dem MyProxy-Service für einen längerfristigen Missbrauch der Privilegien des Nutzers ausreichend.

Der Einsatz der implementierten Delegationsmechanismen ist aufgrund dieser Schwachstellen für einige Nutzergruppen mit hohen Anforderungen in Bezug auf den Schutz der in der Grid-Infrastruktur gespeicherten Daten unmöglich. Zudem kann der unberechtigte Verbrauch von Rechenleistung durch Angreifer zu hohen Kosten für Nutzer führen, deren Proxy-Credentials kompromittiert wurden.

3.3 Beschränkte Delegation von Privilegien in Grid-Infrastrukturen

In Kapitel 3.2 wurden Mechanismen zur Delegation von Nutzerprivilegien in der gLite-Middleware analysiert und bewertet. Dabei erwies sich das alleinige Vertrauen auf die Zugriffskontrollverfahren des auf den Grid-Ressourcen eingesetzten Betriebssystems als nicht gerechtfertigt, um einen wirksamen Schutz der delegierten Privilegien des Nutzers zu gewährleisten. Ein Ansatz, die Folgen einer Kompromittierung von Proxy-Credentials abzumildern, ist die Beschränkung der delegierten Privilegien auf das von den im Namen des Nutzers auszuführenden Aktionen benötigte Minimum.

In der Literatur [LAK+03, PWF+02, ABM+06] werden Ansätze für die beschränkte Delegation von Privilegien in Grid-Infrastrukturen auf GSI-Basis beschrieben, für die teilweise bereits Implementierungen vorliegen. In Kapitel 3.3.2 werden diese vorgestellt und analysiert. Der Fokus der Analyse liegt dabei auf der Effektivität

der Lösungen in Hinblick auf eine verbesserte Sicherheit der delegierten Privilegien vor Missbrauch. Darüber hinaus werden die Effizienz der Ansätze in Bezug auf Komplexität sowie die Unterstützung der Nutzer betrachtet. Um einen neutralen Vergleich der Ansätze zu ermöglichen, werden in Kapitel 3.3.1 Kriterien aufgestellt, mittels derer eine vergleichende Beurteilung in Kapitel 3.3.3 vorgenommen wird.

3.3.1 Kriterien zum Vergleich von Delegationsverfahren

Eine vergleichende Analyse der bestehenden Ansätze hat allgemeingültig und neutral zu erfolgen. Es müssen alle relevanten architektonischen, technischen sowie organisatorischen Aspekte in die Bewertung einbezogen werden. Nachfolgend werden Kriterien aufgestellt, anhand derer die Analyse vorgenommen wird. Diese werden dazu in drei Kategorien unterteilt. In der ersten Kategorie wird die Methodik, die der Beschränkung der Delegation zugrunde liegt, untersucht. Dieser Komplex umfasst die Art der Beschränkung und die Unterstützung von Grid-Diensten sowie die verwendeten Autorisierungsmodelle. Die zweite Kategorie befasst sich mit der Architektur der Ansätze. Diese geben Aufschluss über Komplexität aber auch Flexibilität in der Anwendung. Darüber hinaus wird die Kompatibilität zu GSI-basierten Grid-Infrastrukturen bewertet. In der dritten Kategorie werden die Ansätze in Hinblick auf die Unterstützung der Nutzer untersucht. Die Motivation für diese Kategorie basiert auf der Erkenntnis, dass Sicherheitsmechanismen von Nutzern nur dann akzeptiert und verwendet werden, wenn ihre Anwendung nur geringfügig mehr Komplexität in ihre Arbeitsabläufe bringt, ihre Sicherheit jedoch signifikant erhöht.

3.3.1.1 Methodik

Dimension der Beschränkung Die Beschränkung von Delegationen ist in drei Dimensionen möglich:

- Gültigkeitsdauer der Delegation
- Empfänger der Delegation (*delegatee*)
- Objekt der Delegation und autorisierte Zugriffsart darauf

In GSI-basierten Grid-Infrastrukturen wird lediglich die Gültigkeitsdauer der Delegation beschränkt. Der dafür verwendete Mechanismus ist die Delegation mittels Proxy-Zertifikaten. Dieser ermöglicht dem Empfänger der Delegation, für einen beschränkten Zeitraum im Namen des Nutzers zu agieren. Die mangelnde Sicherheit dieser Art der Beschränkung wurde in den Kapiteln 3.2.1 bis 3.2.4 belegt. Die Beschränkung der Delegation auf definierte Empfänger oder Gruppen von Empfängern ist dann möglich, wenn diese bei Erstellen der Delegation bekannt sind. In den per definitionem dynamischen Grid-Infrastrukturen ist diese Bedingung insbesondere bei der Ablaufplanung von Rechenaufträgen nicht gegeben. Die dritte Dimension

der Beschränkung von Delegationen ist die Autorisierung von Zugriffen bzw. Zugriffsarten auf definierte Objekte des Nutzers. Damit kann die Sicherheit der Objekte des Nutzers vor missbräuchlichem Zugriff wirksam verbessert werden, wenn die in einer Aktion des Nutzers benötigten Objekte nur einer Teilmenge der Objekte entsprechen, auf die er Zugriff hat.

Autorisierungsmodell Die Autorisierung von Nutzern erfolgt in Grid-Umgebungen durch Attributautoritäten, die von den Virtuellen Organisationen direkt oder in deren Auftrag (vgl. Kapitel 2.3.4) betrieben werden. In der Regel kommt ein rollenbasiertes (RBAC) Zugriffskontrollverfahren (vgl. Kapitel 2.3.1) zum Einsatz, das in einigen Fällen mit diskreten Zugriffsrechten (DAC) kombiniert wird (vgl. Kapitel 2.3.4.1). Abhängig vom gewählten Autorisierungsmodell kann die Beschränkung der Delegation durch den Nutzer auf zweierlei Weise erfolgen. Zum Einen kann der Nutzer eine Untermenge der Rollen, in denen er Mitglied ist, für die Delegation aktivieren. Zum Anderen kann er diskrete Rechte spezifizieren und nachfolgend delegieren, über die er durch seine Rollenmitgliedschaften innerhalb der VO verfügt. Der Unterschied zwischen diesen beiden Ansätzen liegt in der Abwägung von Granularität der delegierten Privilegien gegen Aufwand in der Verwaltung von Autorisierungsinformationen. Der diskrete Ansatz verfügt über den Vorteil eines enger spezifizierbaren Umfangs der Delegation, ist jedoch bei zentraler Verwaltung von Objekten aufwändiger als das rollenbasierte Modell. Eine Kombination beider Modelle mittels rollenbasierter Autorisierung des Nutzers durch die Attributautorität sowie einer diskreten Beschränkung der Delegation durch den Nutzer verbindet den Vorteil der weniger komplexen Verwaltung im RBAC-Modell mit der engeren Spezifikation der delegierten Rechte im DAC-Modell.

Objekte der Delegation Die Delegation von Privilegien kann für Objekte auf der Ebene der Grid-Middleware aber auch des Betriebssystems der lokalen Ressource erfolgen. Delegierte Privilegien, die sich auf lokale Ressourcen beziehen, können Zugriffe auf Objekte des Dateisystems, wie einzelne Dateien oder ganze Verzeichnisse, aber auch Zugriffe auf Gerätedateien wie Netzschnittstellen umfassen. Auf der Grid-Ebene lassen sich neben Zugriffsrechten für konkrete Objekte, wie Dateien eines Speichersystems, auch solche für abstrakte Objekte sowie Privilegien zur Ausführung zustandsbehafteter Vorgänge definieren. Darunter fallen Rechenaufträge oder Datentransfers zwischen Speichersystemen. Der Vorteil einer Definition von Zugriffsrechten auf Grid-Ebene ist die erhöhte Abstraktion, die auch die Autorisierung von nicht-atomaren Aktivitäten ermöglicht. Zusätzlich wird der Nutzer somit von der genauen Kenntnis der genutzten Ressourcen entlastet. Die Berücksichtigung des Zustands von Vorgängen für die Zugriffskontrolle ermöglicht zudem die gezielte Delegation von Privilegien für Vorgänge, deren genaue Ausführungsdauer zum Zeitpunkt der Definition der Delegation unbekannt ist.

3.3.1.2 Architektur

Komplexität Ein wichtiger Aspekt bei der Beurteilung der Sicherheit von verteilten Systemen ist deren Komplexität. Diese ergibt sich aus dem Aufbau des Systems, d. h. den einzelnen Komponenten und deren Interaktion. Die Erfahrung zeigt, dass eine hohe Komplexität zu unentdeckten bzw. schwer zu beherrschenden Sicherheitslücken führen kann [Sch04, VM01]. Ziel eines Systems zur beschränkten Delegation muss es daher sein, zusätzliche Komplexität auf das zum Erreichen der funktionalen Ziele minimal erforderliche Maß zu beschränken. Insbesondere gilt es daher, folgende zur Komplexität einer Architektur erheblich beitragenden Faktoren soweit möglich zu vermeiden:

- Einführung zusätzlicher Dienste zur Unterstützung der Delegation. Sind diese darüber hinaus zentral organisiert, stellen sie zudem ohne zusätzliche Redundanzmechanismen eine einzelne Fehlerstelle (*single point of failure*) dar, die auch für Dienstverweigerungsangriffe (*denial of service attack*) ausgenutzt werden kann.
- Einsatz ungeprüfter, bzw. im Grid-Kontext nicht eingesetzter Protokolle zur Kommunikation zwischen den Diensten.
- Einführung zusätzlicher Sprachen zur Formulierung von Richtlinien.

Flexibilität Mit der Flexibilität eines Ansatzes wird dessen Fähigkeit beschrieben, auf wechselnde Gegebenheiten angepasst zu werden. Es müssen folgende Änderungen in der Grid-Infrastruktur berücksichtigt und vom Ansatz zur Delegation unterstützt werden:

- Hinzufügen und Entfernen neuer Objekte. Um den administrativen Aufwand der Verwaltung von Delegationsmechanismen zu begrenzen, sollte der Nutzer neue Objekte selbsttätig hinzufügen können.
- Hinzufügen und Entfernen von Instanzen bereits unterstützter Dienste. In einer dynamischen Grid-Umgebung ist dies ein alltäglicher Vorgang und muss mit geringem Aufwand durchführbar sein.
- Einführung zusätzlicher Dienstarten. Die Unterstützung zusätzlicher Dienstarten stellt eine erhebliche Erweiterung eines Ansatzes dar, muss in der Architektur jedoch prinzipiell vorgesehen werden.

Kompatibilität Die Kompatibilität eines Ansatzes mit bereits in Grid-Verbünden genutzten Datenformaten, Kommunikationsprotokollen und Sprachen zur Formulierung von Richtlinien stellt keineswegs einen Selbstzweck dar. Vielmehr kann auf bereits geprüfte Komponenten zurückgegriffen und somit der Aufbau redundanter Strukturen vermieden werden. Durch einen solchen Ansatz kann zudem die durch den Mechanismus zusätzlich eingeführte Komplexität begrenzt werden. In

Grid-Umgebungen, die auf der gLite-Middleware basieren, leiten sich folgende Fragestellungen ab:

- Werden Proxy-Zertifikate für den Transport der Richtlinien zu den Grid-Ressourcen verwendet? Proxy-Zertifikate stellen das verbreitetste Transportmedium für Delegationsinformation in Grid-Umgebungen dar. Sie lassen sich zudem flexibel durch Zertifikatserweiterungen ergänzen.
- Wird die Autorisierung mittels VO-Attributen unterstützt? Verbleibt die *Source of Authority* (SoA) bei der Virtuellen Organisation, implementiert der Ansatz zur begrenzten Delegation somit nur eine Beschränkung der durch die SoA verliehenen Privilegien bzw. Rollen?

3.3.1.3 Nutzerunterstützung

Sicherheitsmechanismen komplexer IT-Systeme, wie sie auch in Grid-Infrastrukturen Anwendung finden, erfordern in der Regel speziell geschulte Administratoren. Für diese Gruppe ist ein nur mit besonderen Kenntnissen zu bedienendes System akzeptabel. Diese Annahme gilt jedoch nicht für Delegationsmechanismen, die direkt durch Nutzer verwendet werden. Diese verfügen üblicherweise über keinen entsprechenden fachlichen Hintergrund und können ein solches System nur dann erfolgreich nutzen, wenn bei der Konzeption des Verfahrens schon auf die einfache Verwendbarkeit geachtet wurde. Diese Forderung wird schon von Saltzer und Schroeder in [SS75] als psychologische Annehmbarkeit (*psychological acceptability*) von Benutzerschnittstellen erhoben:

> „It is essential that the human interface be designed for ease of use, so that users routinely and automatically apply the protection mechanisms correctly.“

Durch Bishop wird diese Forderung in [Bis02] aufgenommen und generell auf Sicherheitsmechanismen erweitert:

> „The principle of psychological acceptability states that security mechanisms should not make the resource more difficult to access than if the security mechanisms were not present.“

Sind diese Forderungen nach einer nur geringfügig höheren Komplexität in der Bedienung des Systems nicht erfüllt, ist davon auszugehen, dass die Nutzer – soweit möglich – auf die Anwendung der Sicherheitsmechanismen verzichten. Für Delegationsmechanismen bedeutet diese Vermeidungsstrategie, dass Nutzer geneigt sind, einen größeren Teil ihrer Privilegien zu delegieren, als für die Ausführung der gewünschten Aktion notwendig ist. Die Unterstützung, die die untersuchten Delegationsmechanismen für die Nutzer bieten, ist somit für deren Bewertung ein weiteres essentielles Kriterium. Es lassen sich zwei Forderungen nach Funktionalität ableiten, die benutzerfreundliche Delegationsmechanismen erfüllen sollten:

Graphische Benutzeroberfläche Die Erstellung von Richtlinien für die Delegation von Privilegien muss für die Nutzer einfach durchzuführen sein. Dafür bieten sich graphische Benutzeroberflächen an, die den Nutzern ihre zur Verfügung stehenden Privilegien in übersichtlicher Weise zur Auswahl präsentieren. Es sollten Benutzeroberflächen für folgende Funktionalitäten zur Verfügung stehen:

- Auswahl der zu delegierenden Attribute/Rollen.
- Spezifikation diskreter Objekte und autorisierte Zugriffsarten auf diese. Diese Funktionalität kann entfallen, wenn entsprechende Automatismen bereitgestellt werden.
- Autorisierung zustandsbehafteter Vorgänge im Grid. Diese Funktionalität kann entfallen, wenn entsprechende Automatismen bereitgestellt werden.

Automatische Beschränkung Für einige Abläufe benötigt die Grid-Middleware keine zusätzlichen Informationen, um die zu delegierenden Privilegien auszuwählen. In diesen Fällen bieten sich Automatismen an, die Delegationen selbsttätig, d. h. ohne Nutzerinteraktion, beschränken.

3.3.2 Bestehende Ansätze

Die in der Literatur beschriebenen Ansätze werden in diesem Abschnitt anhand der in Kapitel 3.3.1 aufgestellten Kriterien und Anforderungen analysiert. Die Ergebnisse der Analysen sind in Tabelle 3.3 auf Seite 68 zusammengefasst. Um einen Vergleich mit den Eigenschaften der Delegationsmechanismen der gLite-Middleware zu ermöglichen, sind diese dort ebenfalls aufgeführt.

3.3.2.1 Delegationsmechanismen der gLite-Middleware

Die Delegationsmechanismen der gLite-Middleware wurden in den Kapiteln 3.1 und 3.2 analysiert. Eine Zusammenfassung der Eigenschaften ist in Tabelle 3.3 auf Seite 68 dargestellt. Im Einzelnen begründen sich die Beurteilungen wie folgt:

1. Methodik In der gLite-Middleware kommen in ihrer Gültigkeitsdauer (1.1.1) beschränkte Proxy-Zertifikate zum Einsatz. Diese Beschränkung wird durch den Nutzer vollzogen. Die Beglaubigung der Mitgliedschaft innerhalb einer VO wird durch den VOMS ausgestellt und ist in ihrer Gültigkeit ebenfalls beschränkt (1.1.1). Die Empfänger (1.1.2) der Delegation sind weder durch den Nutzer noch durch die VO zu beschränken. Objekte (1.1.3) können indirekt über die Definition von Attributen und Zuweisung dieser zu einzelnen Nutzern adressiert werden. Innerhalb der VO kommt als Autorisierungsmodell RBAC (1.2.1) zum Einsatz, der Nutzer kann

die zu delegierenden Attribute auswählen, jedoch nicht definieren. Die Delegationsmechanismen der gLite-Middleware ermöglichen die Adressierung von Objekten auf Grid-Ebene (1.3.1). Dies kann z. B. die Delegation des Privilegs umfassen, Grid-Ressourcen wie etwa ein definiertes *Computing Element* zu nutzen. Eine feingranulare Adressierung von Objekten wie einzelnen Dateien auf Speicherressourcen ist mit dieser Funktionalität jedoch nicht sinnvoll durchführbar. Die Delegation von Privilegien zur Ausführung zustandsbehafteter Vorgänge (1.3.2) ist nicht möglich.

2. Architektur Die in *gLite* verwendeten Delegationsmechanismen führen den MyProxy-Dienst (2.1.1) zur Unterstützung langlebiger Rechenaufträge ein. Dieser kann mehrfach in der Grid-Infrastruktur vorhanden sein, theoretisch ist ein MyProxy-Server pro Rechenauftrag möglich. Weiterhin wird ein Protokoll (2.1.2) für die Kommunikation zwischen Nutzer und MyProxy-Dienst, bzw. zwischen *proxy-renewd* des WMS und MyProxy-Dienst etabliert. Zusätzliche Policy-Sprachen (2.1.3) werden nicht eingeführt. Das Hinzufügen neuer Objekte (2.2.1) für die beschränkte Delegation durch den Nutzer ist in der gLite-Middleware nicht vorgesehen. Weitere Instanzen bereits vorhandener Dienstearten (2.2.2) werden implizit dadurch hinzugefügt, dass zusätzliche Attribute durch die SoA der VO generiert und den Nutzern zugewiesen werden. Weitere Dienstearten (2.2.3) können durch die Delegationsmechanismen in *gLite* ebenfalls lediglich implizit über die Vergabe von Attributen unterstützt werden, da eine Beschränkung der Delegation direkt durch den Nutzer in der Architektur der gLite-Middleware nicht vorgesehen ist. Zusätzlich müssen entsprechende Zugriffskontrollmechanismen für die Ressourcen erstellt werden. Die Frage der Kompatibilität (2.3) der in der gLite-Middleware zur Authentifizierung, Autorisierung und Delegation von Nutzerprivilegien verwendeten Verfahren ist trivial, da *gLite* nur zum Vergleich hinzugezogen wird.

3. Nutzerunterstützung Die gLite-Middleware sieht keine graphische Benutzeroberflächen zur Unterstützung des Nutzers in der Anwendung der Delegationsmechanismen vor (3.1.1 - 3.1.3). Automatische Beschränkungen der Delegation finden mit einer Ausnahme ebenfalls nicht statt (3.2.1, 3.2.3). Diese Ausnahme betrifft die Delegation der Nutzerprivilegien an den oder die *Worker Nodes*, die den Rechenauftrag des Nutzers ausführen. Das für diese Delegation abgeleitete Proxy-Zertifikat enthält den zusätzlichen Eintrag `"CN=limited proxy"` im DN des Subjektfeldes. Ziel dieses Eintrags ist es, den Zugriff auf Rechenressourcen (3.2.2) zu unterbinden. Dafür wird bei jedem Zugriff auf ein CE das Vorhandensein dieses Eintrags überprüft und im positiven Fall der Zugriff verweigert. Dieser Mechanismus dient dazu, den Schaden im Falle des Missbrauchs eines Proxy-Zertifikates zu begrenzen. Die Wirksamkeit des Verfahrens ist jedoch stark begrenzt, da es nur bei missbräuchlichem Zugriff auf einen WN schützt und auch nur die Ausführung weiterer Rechenaufträge begrenzt. Der Zugriff auf alle weiteren Dienste der Grid-

Infrastruktur ist für die Gültigkeitsdauer des Proxy-Zertifikats im Rahmen der Privilegien des Nutzers gestattet.

3.3.2.2 PRIMA – Privilege Management and Authorization

Das Autorisierungssystem PRIMA [Lor04] wurde mit dem Ziel entwickelt, feingranulare Privilegien in Grid-Infrastrukturen verwalten und durchsetzen zu können. Es basiert auf dem Globus Tollkit 2 und stellt neben dem als Erweiterung des *Grid Resource Allocation Manager* (GRAM) implementierten Zugriffskontrollmoduls Werkzeuge für Nutzer und Administratoren bereit, die die Erstellung von Richtlinien erleichtern sollen. Administratoren von Rechenressourcen werden durch PRIMA in die Lage versetzt, Nutzern Privilegien auf Betriebssystemebene zuzuweisen. Diese ermächtigen die Nutzer zum Zugriff auf lokale Dateien, zum Ausführen von Programmen oder zur Nutzung lokaler Netze. Privilegien können durch den Nutzer an andere Nutzer oder in seinem Auftrag agierende Dienste delegiert werden. PRIMA verhält sich zu den in Kapitel 3.3.1 aufgestellten Kriterien und Forderungen wie folgt (vgl. Tabelle 3.3 auf Seite 68):

1. Methodik PRIMA ermöglicht die Beschränkung der Delegation sowohl in der Gültigkeitsdauer (1.1.1) als auch in Bezug auf die Empfänger der Delegation (1.1.2). Eine Spezifikation von zu delegierenden Privilegien für Zugriffe auf Objekte (1.1.3) ist ebenfalls möglich. Die Vergabe von Privilegien erfolgt im Regelfall nach dem DAC-Ansatz (1.2.2), Administratoren können jedoch auch Gruppen von Nutzern im Kontext einzelner Ressourcen bilden und für diese Gruppen Privilegien vergeben. Diese Funktionalität bildet das RBAC-Modell (1.2.1) nach. Durch das Fehlen einer Attributautorität auf Grid-Ebene, z. B. durch die Implementierung des VO-Modells (2.3.2), ist jedoch eine Gruppenbildung, die für die gesamte Grid-Infrastruktur gültig ist, nicht möglich. PRIMA autorisiert ausschließlich Zugriffe auf lokale Objekte von GRAM-Ressourcen. Die Beschränkung der Delegation auf Objekte (1.3.1) oder zustandsbehaftete Vorgänge (1.3.2) der Grid-Ebene ist nicht vorgesehen.

2. Architektur Die Architektur von PRIMA zeichnet ein Verzicht auf zentrale Dienste (2.1.1) aus. Komponenten von PRIMA werden ausschließlich lokal auf den Ressourcen sowie auf den Rechnern der Nutzer eingesetzt. Durch den Transport der Autorisierungsinformationen in Erweiterungen von Proxy-Zertifikaten (2.3.1) kann zudem auf zusätzliche Kommunikationsprotokolle verzichtet werden (2.1.2). Die durch PRIMA verwendete Policy-Sprache (2.1.3) XACML wird im Grid-Umfeld bereits genutzt, durch die Nutzung eines verbreiteten Standards wird zusätzliche Komplexität vermieden. Die Delegation von Privilegien, die den Zugriff auf Objekte (2.2.1) autorisieren, erfolgt durch den Nutzer ohne Unterstützung durch das Autorisierungssystem. Der Nutzer muss dafür über genaue Informationen bezüglich der

ihm zugänglichen Objekte auf den Ressourcen verfügen. Das Zugriffskontrollmodul von PRIMA ist spezialisiert auf die Autorisierung von Zugriffen auf lokale Objekte von GRAM-Diensten, das Hinzufügen weiterer Dienste dieses Typs (2.2.2) ist möglich. Eine Unterstützung weiterer Dienstearten (2.2.3), insbesondere solcher, die auf Grid-Ebene arbeiten, ist jedoch nicht in der Architektur vorgesehen.

3. Nutzerunterstützung PRIMA beinhaltet eine graphische Benutzeroberfläche (*Privilege Creator*) für die Spezifikation der zu delegierenden Privilegien. Dabei wird dem Nutzer keine Auswahl der zur Verfügung stehenden Objekte, Attribute und Rollen (3.1.1) geboten, die Spezifikation erfolgt manuell. Eine Überprüfung der spezifizierten Delegation auf Korrektheit der Syntax und Plausibilität findet nicht statt. Da die Delegationsmechanismen von PRIMA lediglich lokale Objekte auf GRAM-Ressourcen unterstützt, sind Oberflächen für weitere Funktionalitäten nicht vorgesehen (3.1.2, 3.1.3). Eine automatische Beschränkung von Delegationen findet ebenfalls keine Anwendung (3.2.1 - 3.2.3).

3.3.2.3 CAS – Community Authorization Service

Der *Community Authorization Service* [Glo06, PWF+02, PWF+03] erfüllt im *Globus Toolkit* eine konzeptionell mit der des VOMS in der gLite-Middleware vergleichbare Rolle. Mittels CAS können Virtuelle Organisationen verwaltet und deren Mitgliedern Rollen sowie Privilegien zugewiesen werden. Um den Ressourcen gegenüber ihre Privilegien nachweisen zu können, erhalten die Nutzer auf Anfrage signierte Beglaubigungen (*assertions*). Diese Beglaubigungen werden in Proxy-Zertifikate eingebettet und im Push-Verfahren zur Ressource übertragen. Der wichtigste Unterschied zum VOMS besteht in der Eigenschaft des CAS, neben den Informationen über die Rolle des Nutzers innerhalb der VO auch diskrete Privilegien des Nutzers verwalten zu können. Die Eigenschaften des CAS (vgl. Tabelle 3.3 auf Seite 68) in Bezug auf die in Kapitel 3.3.1 aufgestellten Kriterien begründen sich wie folgt:

1. Methodik Die Beschränkung der Delegation erfolgt konzeptionell ähnlich wie in der gLite-Middleware. Abweichend davon unterstützt der CAS die Delegation von Privilegien für diskrete Objekte auf Ressourcen (1.1.3). Im Unterschied zum Ansatz von PRIMA werden diese Privilegien jedoch zentral auf dem CAS durch das VO-Management verwaltet. CAS unterstützt sowohl die Zuordnung von Nutzern zu Rollen, als auch die Zuweisung von diskreten Privilegien zu Nutzern (1.2.1,1.2.2). Die beschränkte Delegation von Privilegien auf Grid-Ressourcen (1.3.1) ist möglich, diese sind jedoch auf Objekte des GridFTP-Servers beschränkt. Weitere Ressourcenarten werden nicht unterstützt. Die Delegation von Privilegien in Bezug auf zustandsbehaftete Vorgänge (1.3.2) wird nicht unterstützt.

2. Architektur CAS basiert auf zentralen Diensten (2.1.1). Der CAS bildet für jede VO eine einzelne Fehlerstelle. Die aktuelle Version des CAS basiert auf standardisierten Protokollen und Policy-Sprachen wie *WebServices* und der *Security Assertion Markup Language* (SAML) [Org07b]. Die Flexibilität von CAS in Bezug auf beschränkte Delegation ist aufgrund der zentralisierten Architektur eingeschränkt. Der Nutzer kann nicht selbsttätig Objekte für Delegationen (2.2.1) hinzufügen, auch eine Unterstützung zusätzlicher Dienstearten ist nur mit grundlegenden Änderungen einzuführen (2.2.3). Das Hinzufügen weiterer bereits unterstützter Dienste (2.2.2) ist hingegen einfach zu realisieren. CAS verwendet Proxy-Zertifikate (2.3.1) und implementiert das VO-Modell (2.3.2).

3. Nutzerunterstützung CAS bietet weder eine graphische Benutzeroberfläche (3.1.x) für den Nutzer noch automatische Beschränkungen der delegierten Privilegien (3.2.x) auf das für die jeweilige Aufgabe minimal Notwendige.

3.3.2.4 On-Demand Restricted Delegation

In [ABM+06] wird eine Architektur für dynamische, beschränkte Delegationen von Privilegien in Grid-Infrastrukturen vorgeschlagen. Für eine Analyse des Ansatzes anhand der in Kapitel 3.3.1 aufgestellten Kriterien fehlen jedoch detaillierte Informationen sowie eine veröffentlichte Implementierung. Aus diesem Grund werden an dieser Stelle lediglich die Kernpunkte des Ansatzes vorgestellt:

Dem Ansatz liegt die zentrale These zugrunde, dass der Nutzer vor dem Absenden eines Rechenauftrags nicht in der Lage ist, die dafür benötigten Privilegien zu spezifizieren. Daher sieht die Architektur die Verwendung verteilter Delegationdienste vor, die exklusiv Nutzern oder Ressourcen zur Verfügung stehen und der dynamischen Gewährung von Delegationen dienen. Basis der auf Ontologien basierenden Entscheidungen der Delegationsdienste sind die von den Nutzern vor Beginn des Rechenauftrags hinterlegten Richtlinien für die Delegation ihrer Privilegien. Für die Abfrage des individuellen Privilegienbedarfs der Rechenaufträge bei den Delegationsdiensten der Ressourcen wird die Verwendung von Sprachen aus dem Bereich des *Semantic Web* [KFJ03, TBKM05] vorgeschlagen. Dies gilt auch für die Abfragen zur Erlangung dieser Privilegien bei den Delegationsdiensten der Nutzer.

Ein Nebeneffekt der Verwendung von Delegationsdiensten ist die Möglichkeit, einen Sperrmechanismus für Proxy-Zertifikate zu realisieren. Dieser wird durch die Registrierung von Proxy-Zertifikaten beim jeweiligen Delegationsdienst des Nutzers implementiert. Die Sperrung nimmt der Nutzer durch Löschen der Registrierung vor. Dadurch erhalten Ressourcen, die das entsprechende Proxy-Zertifikat verwenden, keine weiteren Delegationen. Die Sperrung eines Proxy-Zertifikats ist immer dann von Vorteil, wenn der Nutzer noch während dessen Gültigkeitsdauer von der Kompromittierung Kenntnis erhält.

3.3.3 Fazit

In Kapitel 3.3.2 wurden neben den Delegationsmechanismen der gLite-Middleware drei weitere Ansätze analysiert, die eine beschränkte Delegation von Nutzerprivilegien in Grid-Infrastrukturen realisieren. Diese werden nachfolgend anhand der in Kapitel 3.3.1 aufgestellten Kriterien bewertet. Auf eine detaillierte Bewertung des in [ABM+06] vorgestellten Ansatzes wird verzichtet, da bisher noch keine darüber hinausgehenden Informationen sowie keine Implementierung vorliegen. Insgesamt erscheint die hohe Komplexität des Ansatzes in Abwägung des daraus abgeleiteten Nutzens schlecht begründet. Kern des Ansatzes ist die Annahme, dass es Nutzern nicht möglich ist, vor Abgabe eines Auftrags an die Grid-Infrastruktur statische Richtlinien für die Delegation von Privilegien zu spezifizieren. Diese Annahme wird von den Autoren nicht begründet. Es wird jedoch vorausgesetzt, dass es den Nutzern möglich ist, Richtlinien zu spezifizieren, die die Delegationsdienste als Grundlage für Entscheidungen über Delegationsgesuche der Ressourcen verwenden können. Es erscheint zweifelhaft, dass beide Annahmen zutreffen können. Die durch die gLite-Middleware sowie PRIMA und CAS implementierten Delegationsmechanismen verhalten sich in Bezug auf die aufgestellten Kriterien und daraus abgeleiteten Anforderungen wie folgt:

1. Methodik Die Forderung in Kapitel 3.3.1.1, nach der die Autorisierung mittels RBAC durch die Attributautorität der Virtuellen Organisation (1.2.1), die Beschränkung der Delegation jedoch durch den Nutzer für diskrete Objekte (1.2.2) erfolgen sollte, unterstützt keiner der untersuchten Ansätze. PRIMA ermöglicht dem Nutzer zwar die Beschränkung der Delegation auf diskrete Objekte, die Autorisierung für den Zugriff auf diese muss dafür jedoch im Vorhinein durch die AA für jede Ressource separat erfolgen. Die Delegation von Zugriffsrechten für Objekte auf der Grid-Ebene (1.3.1) wird nicht (PRIMA) oder nur unvollständig (*gLite*, CAS) für einzelne Dienste unterstützt. PRIMA unterstützt ausschließlich lokale Objekte auf GRAM-Ressourcen, der Nutzer hat entsprechend über Kenntnisse der lokalen Objekte auf den Ressourcen zu verfügen. Die Delegation von Privilegien für die Ausführung zustandsbehafteter Vorgänge (1.3.2) unterstützt keiner der Ansätze. Die durch das Grid-Paradigma eingeführte Abstraktionsebene wird durch diese Defizite signifikant geschwächt.

2. Architektur Die Definition von diskreten Objekten, für die Zugriffsrechte delegiert werden können, erfolgt in allen untersuchten Ansätzen durch die Attributautorität. Der Aufwand für die Administration der VO wird dadurch erhöht, die Flexibilität für den Nutzer (2.2.1) entsprechend gesenkt, zudem wird der Nutzen aus der Implementierung des RBAC-Modells reduziert. PRIMA implementiert weiterhin nicht das VO-Paradigma (2.3.2), wodurch die Bildung von Nutzergruppen so-

wie die dynamische Zuweisung von Nutzungsrechten zu diesen signifikant erschwert wird.

3. Nutzerunterstützung Die Unterstützung des Nutzers bei der Beschränkung von Delegationen durch graphische Benutzeroberflächen (3.1) ist nur in PRIMA rudimentär gegeben. Der Nutzer benötigt jedoch auch bei Unterstützung durch dieses Werkzeug detaillierte Kenntnisse, wodurch eine Erleichterung der Aufgabe kaum gegeben ist. Eine automatische Beschränkung der Delegationen des Nutzers implementiert keines der untersuchten Verfahren zufriedenstellend. Die Beschränkung der Delegation für Rechenaufträge der gLite-Middleware (vgl. Kapitel 3.3.2.1) ist nicht zielführend, da die Beschränkung erst nach Verlassen der Einflusssphäre des Nutzers eingeführt wird und nur auf einem Dienst wirksam ist. Die Implementierung von wirksamen Funktionalitäten zur Unterstützung des Nutzers ist somit bei keinem Verfahren gegeben, sie ist jedoch eine unabdingbare Voraussetzung für die Akzeptanz durch den Nutzer.

Eigenschaften		gLite	PRIMA	CAS
1. Methodik	**1.1 Dimension der Beschränkung (in Klammern: ausführende Entität)**			
	1.1.1 Gültigkeitsdauer	+ (VO,Nutzer)	+ (Nutzer)	+ (VO,Nutzer)
	1.1.2 Empfänger	−	+ (Nutzer)	−
	1.1.3 Objekt	∘ (Nutzer)	+ (Nutzer)	+ (Nutzer)
	1.2 Autorisierungsmodell (in Klammern: autorisierende Entität)			
	1.2.1 RBAC	+ (VO)	∘ (AA)	+ (VO)
	1.2.2 DAC	−	+ (AA)	∘ (VO)
	1.3 Objekte der beschränkten Delegation			
	1.3.1 Objekte auf Grid-Ebene	∘	−	∘
	1.3.2 Zustandsbehaftete Vorgänge im Grid	−	−	−
2. Architektur	**2.1 Komplexität**			
	2.1.1 zentrale Dienste	∘	+	∘
	2.1.2 Protokolle	∘	+	+
	2.1.3 Policy-Sprachen	+	+	+
	2.2 Flexibilität			
	2.2.1 Objekte	−	∘	−
	2.2.2 Dienste	+	+	+
	2.2.3 Dienstearten	−	−	−
	2.3 Kompatibilität			
	2.3.1 Proxy-Zertifikate	+	+	+
	2.3.2 VO-Autorisierung	+	−	+
3. Nutzerunterstützung	**3.1 Funktionalität der Graphischen Benutzeroberfläche: Auswahl von**			
	3.1.1 Attributen/Rollen	−	∘ (manuell)	−
	3.1.2 Objekten auf Speicherressourcen	−	−	−
	3.1.3 zustandsbehafteten Vorgängen	−	−	−
	3.2 Automatische Beschränkung von Delegationen			
	3.2.1 Zugriffe auf Grid-Speicherressourcen	−	−	−
	3.2.2 Zugriffe auf Grid-Rechenressourcen	∘ (nur für WN)	−	−
	3.2.3 für zustandsbehaftete Vorgänge	−	−	−

Tabelle 3.3: Eigenschaften von Ansätzen zur beschränkten Delegation von Privilegien

4 Lösungsansatz

In Kapitel 3.2 wurde gezeigt, dass eine Kompromittierung von Proxy-Credentials in Grid-Infrastrukturen, die auf der GSI, speziell aber auf der gLite-Middleware basieren, möglich ist. Sowohl die gegenüber ihren Virtuellen Organisationen für ihre Handlungen haftenden Nutzer, als auch die Betreiber von Grid-Ressourcen müssen daher ein vitales Interesse daran haben, den aus Kompromittierungen von Proxy-Credentials resultierenden Missbrauch zu beschränken.

Das Interesse der Nutzer liegt in der Vermeidung von in ihrem Namen missbräuchlich in Anspruch genommenen Leistungen der Grid-Infrastruktur, da diese zu einer Belastung der individuellen Nutzungskontingente sowie weitergehenden juristischen Konsequenzen führen könnten. Die gezielte Beschränkung der delegierten Privilegien durch die Nutzer auf die für die Ausführung der gewünschten Aktionen benötigte Untermenge stellt ein probates Mittel hierfür dar.

Damit ein Missbrauch der Grid-Ressourcen wirksam verhindert werden kann, müssen die Richtlinien zur Beschränkung der Delegation auf den Ressourcen durchgesetzt werden. Dies liegt im Interesse der Ressourcenanbieter, da eine Abrechnung von Leistungen unter Verwendung hierfür vom Nutzer nicht berechtigter Credentials in der Regel unmöglich sein dürfte. Auch datenschutzrechtliche Haftungsfragen im Fall von unrechtmäßig offengelegten Daten der Nutzer können für die Ressourcenanbieter von Bedeutung sein.

Die in Kapitel 3.3 durchgeführte Analyse von Ansätzen zur Delegation von Nutzerprivilegien hat ergeben, dass keiner dieser Ansätze die Anforderungen der aufgestellten Kriterien erfüllt. Ein wirksamer Schutz der zur Delegation verwendeten Proxy-Credentials besteht nicht oder nur für wenige Anwendungsbereiche. Daher wurde – ausgehend von den in Kapitel 3.3.3 zusammengefassten Erkenntnissen – ein neuer, umfassender Ansatz für die beschränkte Delegation von Privilegien erarbeitet, der in diesem Kapitel vorgestellt wird. Dieser Ansatz bildet auch die Grundlage für die in Kapitel 5 vorgestellte Implementierung am Beispiel der gLite-Middleware.

4.1 Vorüberlegungen und Zielsetzungen

GSI-basierte Middlewares wie das *Globus Toolkit* und die gLite-Middleware bilden die Grundlage für die meisten der heute betriebenen Grid-Infrastrukturen. Ausgehend von deren Verbreitung in europäischen Grid-Projekten sowie der lokal vorhandenen Erfahrungen (vgl. Anhang A) empfahl es sich, die gLite-Middleware als Basis

für den Entwurf eines umfassenden Ansatzes zur beschränkten Delegation zu verwenden. Ein Einsatz in Grid-Infrastrukturen auf Basis des *Globus Toolkit* sollte dabei zu einem späteren Zeitpunkt möglich und nicht von vornherein durch Designentscheidungen ausgeschlossen werden. Erleichtert wird eine solche Portierung durch den Umstand, dass beide Middlewares durch die Nutzung von Authentifizierungs- und Autorisierungsmechanismen der GSI in diesem Bereich prinzipiell kompatibel sind, auch wenn sich die durch diese Mechanismen geschützten Dienste unterscheiden.

Das nachträgliche Einführen von Sicherheitsmechanismen in etablierte Systeme ist prinzipiell mit besonderen Anforderungen verbunden. Eine dieser Anforderungen ist die Wahrung der Kompatibilität mit dem System, um die Funktion der bestehenden Komponenten nicht zu gefährden. Eine andere ist die dazu unter Umständen in Konflikt stehende Sicherstellung der intendierten Funktion des neuen Mechanismus.

Ausgehend von diesen Überlegungen und den Anforderungen in Kapitel 3.3.1 ergeben sich folgende Zielsetzungen:

- Die Kompatibilität zu den implementierten Authentifizierungs- und VO-basierten Autorisierungsmechanismen ist zu wahren. Auch sind bestehende Standards, wie Protokolle und Sprachen, wo möglich zu befolgen, um die spätere Portierung auf andere Middlewares zu vereinfachen.
- Der Umfang der delegierten Privilegien muss effektiv eingeschränkt werden, um das Risiko eines Missbrauchs signifikant zu senken. Unbeschränkte Delegationen dürfen nicht an Entitäten außerhalb der Einflusssphäre des Nutzers bzw. seiner Institution erteilt werden.
- Die Komplexität und der Aufwand der Verwaltung von Nutzerprivilegien und Zugangskontrollmechanismen auf VO- bzw. Site-Ebene darf nicht signifikant steigen, da andernfalls eine Skalierung auf große Nutzerzahlen erschwert würde.
- Neben statischen Grid-Objekten, wie Dateien und Verzeichnissen auf *Storage Elements*, sind auch dynamische, zustandsbehaftete Vorgänge wie Rechenaufträge für eine Delegation zu unterstützen. Weiterhin muss eine Erweiterung des Delegationsmechanismus für zusätzliche Dienste mit geringem Aufwand möglich sein.
- Der Nutzer muss in der Anwendung der Delegationsmechanismen durch geeignete Mittel effektiv unterstützt werden. Die Erfahrung zeigt, dass Sicherheitsmechanismen von Nutzern nur akzeptiert werden, wenn deren Anwendung mit nur geringem Mehraufwand verbunden, der daraus resultierende Sicherheitsgewinn aber erheblich ist. Diese Mittel können aus graphischen Benutzeroberflächen oder Mechanismen zur Automatisierung von Vorgängen bestehen.

4.2 Methodik

4.2.1 Delegation durch Selbstbeschränkung der Nutzer

Die Vergabe von Privilegien innerhalb heutiger Grid-Verbünden ist hierarchisch organisiert (vgl. Abbildung 4.1). Ressourcenbetreiber erteilen den in dieser Umgebung etablierten Virtuellen Organisationen Nutzungsrechte über die Vereinbarung von *Service Level Agreements* und *Acceptable Use Policies* (vgl. Kapitel 2.3.4). Die Virtuellen Organisationen geben diese Nutzungrechte in Gänze oder als Teilmenge an ihre Mitglieder weiter. Dies erfolgt nach dem um generische Attribute ergänzten Modell der rollenbasierten Zugriffskontrolle in Form von VO-Attributen, d. h. signierten Beglaubigungen über Mitgliedschaften in Untergruppen der VO sowie Rollen oder generischen Attributen im Kontext der VO.

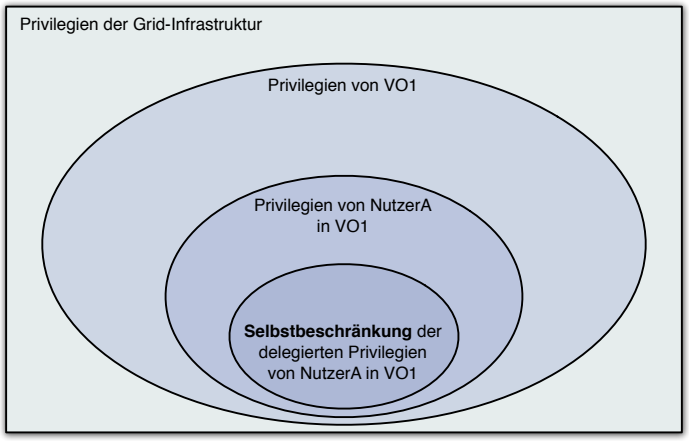


Abbildung 4.1: Hierarchie der Privilegien in Grid-Infrastrukturen

Die gleichzeitige Mitgliedschaft von Nutzern in multiplen Virtuellen Organisationen ist prinzipiell möglich. In den AUP der Virtuellen Organisationen und der Ressourcenbetreiber muss jedoch klar geregelt werden, ob die daraus abgeleiteten Privilegien gemeinschaftlich für atomare Aktionen im Grid ausgeübt werden dürfen.

Die bestehenden Mechanismen der etablierten Grid-Middlewares ermöglichen die gezielte Delegation einer Teilmenge von Privilegien nur in eingeschränkter Form durch in grober Granularität delegierte Privilegien zur Ausübung der eigenen Gruppen- oder Rollenmitgliedschaften innerhalb der VO. Nutzern ist es möglich, eine Teilmenge der eigenen VO-Attribute zu delegieren. Auf die Granularität der Attribute oder die konkret aus diesen abgeleiteten Rechte haben sie jedoch keinen Einfluss.

Der in dieser Arbeit vorgestellte Ansatz erhält die bestehende Autorisierungsarchitektur und ergänzt sie um einen Mechanismus, der eine fein-granulare Delegation von Privilegien der Nutzer an Grid-Ressourcen ermöglicht. Der Ansatz führt Autorisierungsrichtlinien ein, in denen Nutzer Regeln erlassen können. In diesen werden Teilmengen der den Nutzern durch die Attributautoritäten zugewiesenen Privilegien spezifiziert sowie das Recht zu ihrer Ausübung an Grid-Ressourcen delegiert, die ihre Aufträge bearbeiten.

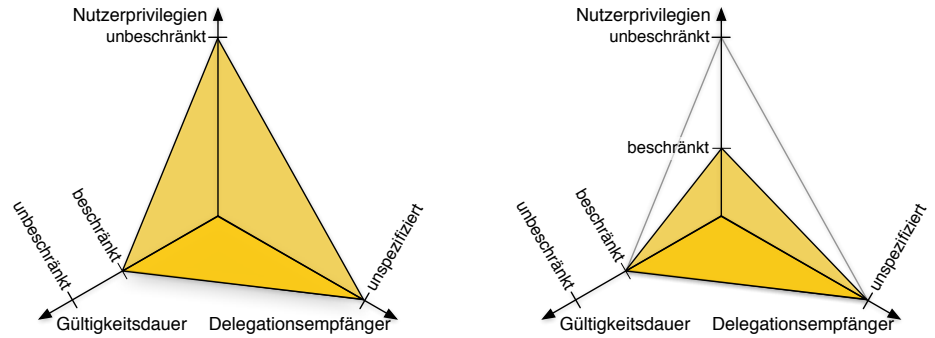
Zu der in Formel 4.1 dargestellten bisherigen Form der Delegation kommt damit die zusätzliche Komponente der nutzerdefinierten Richtlinie hinzu, die in diesem Fall eine Regel beinhaltet (vgl. Formel 4.2).

$$keyEEC_U \quad \textbf{delegates} \quad isMember((U\,\textbf{in}\,R)_{VO_x})^{\wedge}* \quad \textbf{to} \quad keyPC_U \tag{4.1}$$

$$keyEEC_U \quad \textbf{delegates} \quad excercise(privilege) \quad \textbf{to} \quad keyPC_U \tag{4.2}$$

In Grid-Infrastrukturen ist diese Art der nutzerdefinierten Richtlinien als Selbstbeschränkung der den Nutzern von Virtuellen Organisationen zugesicherten Privilegien zu interpretieren. Dieser Umstand erklärt sich aus der Tatsache, dass Nutzer über keine eigenständige Autorität in der Grid-Infrastruktur verfügen. Davon ausgehend ist es für Nutzer unmöglich, mittels der von ihnen erstellten Richtlinien die Menge an Privilegien zu überschreiten, die ihnen durch die Attributautoritäten der Virtuellen Organisationen zugewiesen wurde. Abbildung 4.1 auf der vorherigen Seite veranschaulicht diesen Umstand. Je nach Ausgestaltung der AUP zwischen Grid-Infrastruktur und VO, bzw. zwischen VO und Nutzer, können Nutzer jedoch Privilegien multipler Virtueller Organisationen gemeinsam in nutzerdefinierten Richtlinien für eine Aktion aktivieren.

Die Methode der Selbstbeschränkung der delegierten Privilegien durch die Nutzer hat Vorteile gegenüber Ansätzen, die auf zentral organisierten Delegationsmechanismen beruhen. So steigt der Aufwand zur Verwaltung von Virtuellen Organisationen durch eine fein-granulare Delegation von Nutzerprivilegien nicht an. Die Verantwortlichen einer VO nehmen weiterhin nur eine grobe Klassifizierung der Nutzer in Form einer Zuweisung von Attributen wie Gruppenmitgliedschaften oder Rollen vor. Weiterhin sinkt das mit einem Missbrauch der Delegation verbundene Risiko, da Nutzer genauere Richtlinien für die Delegation erstellen können, als dies mit vertretbarem Aufwand durch die Attributautorität der VO möglich wäre. Dieser Umstand ergibt sich vornehmlich aus der Tatsache, dass die Nutzer genaue Kenntnis über die für ihre Aktionen im Grid benötigten Objekte haben. Mit diesem Wissen lassen sich Richtlinien spezifizieren, deren enthaltene Regeln näher am Minimum der für die Aktionen benötigten Privilegien liegen.



(a) Beschränkung der Gültigkeitsdauer

(b) Zusätzliche Beschränkung der delegierten Nutzerprivilegien

Abbildung 4.2: Beschränkung von Delegationen

4.2.2 Dimensionen der Beschränkung

Die Delegation von Privilegien kann (vgl. Kapitel 3.3.1.1) in drei Dimensionen beschränkt werden. Im einfachsten Fall wird die Gültigkeitsdauer der Delegation (vgl. Abbildung 4.2a) limitiert. Damit ist der Besitzer der Delegation nach Ablauf der Gültigkeitsdauer nicht mehr berechtigt, die delegierten Privilegien auszuüben. Dieser Ansatz wird in Grid-Infrastrukturen, die auf der GSI basieren, durch den Einsatz von Proxy-Zertifikaten verfolgt. Dass dieser Ansatz allein nicht ausreicht, um Delegationen hinreichend zu sichern, wurde bereits in Kapitel 3.2 dargelegt. Insbesondere verhindert er nicht das Offenlegen von Daten des Nutzers, die auf Grid-Ressourcen gespeichert werden.

Der in dieser Arbeit vorgestellte Ansatz kombiniert die Mechanismen der GSI mit einer zusätzlichen Dimension der Beschränkung. Im Gegensatz zu den Delegationsmechanismen der GSI, in der Empfänger von Delegationen das Recht zur Ausübung des vollen Umfangs der Privilegien des Nutzers erhalten, wird nun eine fein-granulare Beschränkung der durch die Attributautorität der VO zugewiesenen Nutzerprivilegien ermöglicht (vgl. Abbildung 4.2b). Die Beschränkung der Nutzerprivilegien erfolgt durch Aufstellen von diskreten Regeln, die in nutzerdefinierten Richtlinien zusammengefasst werden.

Eine Beschränkung der Delegation auf zur Nutzung berechtigte Entitäten (Delegationsempfänger) erscheint in Grid-Infrastrukturen nicht zielführend und wird deshalb in diesem Ansatz nicht durchgeführt. Das größte Hindernis bei der statischen Beschränkung der Delegation auf Ressourcen ist die hohe Dynamik in Grid-Infrastrukturen. So ist es üblich, dass dem Nutzer im vorhinein nicht bekannt ist, welche Rechenressource für einen bestimmten Auftrag verwendet wird, so dass eine Spezifikation von Richtlinien vor Beauftragung unmöglich ist. Auch ist es möglich, dass Rechenaufträge wegen eines Fehlers der ausführenden Ressource abbrechen und

deshalb von der Grid-Infrastruktur ohne Benachrichtigung des Nutzers auf einer anderen noch einmal ausgeführt werden. Dies wäre bei einer statischen Beschränkung der Delegation auf bestimmte Ressourcen unmöglich. Eine ähnliche Problematik gilt für Speicherressourcen. Diese nehmen bei Bedarf automatische Replikationen von Dateien vor, um entweder häufig benötigte Dateien in besserer Lokalität zu den diese bearbeitenden Rechenressourcen zu haben oder um die Verfügbarkeit im Fehlerfall zu verbessern. Bei einer statischen Beschränkung der Delegation auf den Zugriff auf eine Speicherressource wäre der Zugriff auf die dynamisch ausgesuchte unter Umständen untersagt und der entsprechende Auftrag würde scheitern.

4.2.3 Objektklassen

In Kapitel 3.3.1.1 wurden die in Grid-Infrastrukturen verwendeten Klassen von Objekten im Nutzerzugriff vorgestellt. Entsprechend der dort aufgestellten Forderung unterstützt der vorgelegte Ansatz Objekte auf Ebene der Grid-Middleware. Lokale Objekte auf Betriebssystemebene der Grid-Ressourcen sollten lediglich temporär für die Dauer von Rechenaufträgen im Besitz des Nutzers sein und sind daher für diesen von nur transientem Interesse. Die Objekte der Grid-Ebene gliedern sich in statische und zustandsbehaftete, also veränderliche, Objekte.

4.2.3.1 Statische Objekte

Statische Objekte sind entweder existent oder nicht existent und haben somit für die Delegation von sie betreffenden Privilegien lediglich einen relevanten Zustand. Sie werden innerhalb der Grid-Infrastruktur durch einen eindeutigen Bezeichner adressiert. In heutigen Grid-Infrastrukturen wird diese Objektklasse von Dateien und Verzeichnissen auf Speicherressourcen repräsentiert. Eine Anpassung des Ansatzes auf zukünftige Dienste, deren Objekte über eine ähnliche Charakteristik verfügen, ist mit geringem Aufwand möglich.

Auf Objekte der Datendienste kann auf verschiedene Arten zugegriffen werden, die durch den Ansatz zur beschränkten Delegation unterstützt werden müssen. Typisch sind die Zugriffsarten Lesen (*read*), Schreiben (*write*) sowie Löschen (*delete*). Hinzu kommt bei einigen Implementierungen von Speicherressourcen noch die Zugriffsart des Erstellens von Objekten (*write-once* oder *create*). Die Informationen über die dem Nutzer zugeordneten Privilegien bezüglich dieser Objekte werden in gLite-Infrastrukturen im *File Catalog* (vgl. Kapitel 2.1.3.2) verwaltet.

Die gezielte Beschränkung der Delegation von Privilegien für den Zugriff auf statische Objekte erfolgt durch Einbinden von Regeln in die nutzerdefinierten Richtlinien. Diese Regeln beinhalten die durch den Nutzer autorisierten Zugriffsarten sowie eine Referenz auf das Objekt. Die Zugriffskontrolle auf Grid-Ressourcen anhand

der nutzerdefinierten Richtlinien erfolgt durch Vergleich der in dieser autorisierten Zugriffe mit dem vorliegenden Zugriff.

4.2.3.2 Zustandsbehaftete Objekte

Zustandsbehaftete Objekte werden in Grid-Infrastrukturen durch Vorgänge wie Rechenaufträge oder Datentransfers zwischen Speicherressourcen repräsentiert. Sie werden durch den Nutzer beauftragt, verfügen somit über einen Besitzer und werden mit einem eindeutigen Bezeichner versehen. Die zuständigen Komponenten der Grid-Middleware, im Fall von Rechenaufträgen in gLite-Infrastrukturen der LB-Service (vgl. Kapitel 2.1.3.4) sowie der *File Transfer Service* (FTS) für Dateitransfers, verwalten Informationen über den Zustand dieser Objekte.

Die Delegation von Privilegien für die Ausführung dieser Art von Vorgängen erfolgt durch Einbinden von Regeln in die nutzerdefinierten Richtlinien. Diese Regeln haben die Zugriffsart *execute*, können aber auch genutzt werden, um das Abbrechen (*cancel*) sowie das temporäre Unterbrechen und die spätere Wiederaufnahme (*suspend*, *resume*) von Vorgängen zu autorisieren. Das Objekt wird in diesen Regeln mittels seines eindeutigen Bezeichners referenziert.

Die Zugriffskontrolle anhand der nutzerdefinierten Richtlinien auf den Ressourcen kann durch Überprüfen der Existenz eines solchen Objekts in der Datenbank des verwaltenden Dienstes erfolgen, womit der Wille des Besitzers zur Durchführung dieses Vorgangs prinzipiell belegt ist. Zusätzlich kann der Zustand eines Vorgangs, der sich über seine Lebensdauer verändert, zur Zugangskontrolle genutzt werden. Durch Abfragen des momentanen Zustands eines Vorgangs bei der Zugriffskontrolle kann dadurch z. B. das wiederholte Durchführen (*replay attack*) bereits abgeschlossener oder bereits in Durchführung befindlicher Aufträge unterbunden werden.

4.2.4 Beispiel: Beschränkte Delegation mit nutzerdefinierten Richtlinien

Die Mitgliedschaft eines Nutzers U in VO_x in der Rolle R_1 gibt ihm durch entsprechende Vereinbarungen zwischen VO und Ressourcenbetreiber auf das Verzeichnis `/VOx/R1/` eines *Storage Elements* Rechte zum Schreiben, Lesen sowie Löschen von Dateien. Weiterhin erhält er das Recht, Rechenaufträge an die Grid-Infrastruktur abzugeben. Für einen Rechenauftrag benötigt er die Dateien `/VOx/R1/Data1` und `/VOx/R1/Data2` als Eingabedateien sowie `VOx/R1/Out1` als Ausgabedatei. Für den Rechenauftrag vergibt Nutzer U zusätzlich einen eindeutigen Bezeichner Job_{U1} und registriert diesen auf dem zuständigen LB-Service. Um diesen Rechenauftrag mit den benötigten Privilegien in seinem Namen auszustatten, delegiert der Nutzer seine Privilegien an den geheimen Schlüssel eines Proxy-Zertifikats (vgl. Formel 4.3).

$$keyEEC_U \quad \textbf{delegates} \quad isMember((U\,\textbf{in}\,R_1)_{VO_x})^{\wedge}* \quad \textbf{to} \quad keyPC_U \tag{4.3}$$

Um den möglichen Missbrauch mit seinen Daten auf die für den Rechenauftrag benötigten Privilegien zu beschränken, erlässt der Nutzer eine nutzerdefinierte Richtlinie, die andere als die erlaubten Zugriffe auf Daten untersagt (vgl. Formeln 4.4 bis 4.6). Um das missbräuchliche Abgeben weiterer Rechenaufträge, die sein Kontingent an Rechenzeit belasten würden, zu untersagen, fügt der Nutzer zusätzlich eine weitere Regel (vgl. Formel 4.7) ein, die explizit nur das Ausführen des Rechenauftrags mit dem Bezeichner Job_{U1} gestattet.

$$keyEEC_U \quad \textbf{delegates} \quad read(/VOx/R1/Data1) \quad \textbf{to} \quad keyPC_U \tag{4.4}$$
$$keyEEC_U \quad \textbf{delegates} \quad read(/VOx/R1/Data2) \quad \textbf{to} \quad keyPC_U \tag{4.5}$$
$$keyEEC_U \quad \textbf{delegates} \quad write(/VOx/R1/Out1) \quad \textbf{to} \quad keyPC_U \tag{4.6}$$
$$keyEEC_U \quad \textbf{delegates} \quad execute(Job_{U1}) \quad \textbf{to} \quad keyPC_U \tag{4.7}$$

4.3 Nutzerdefinierte Autorisierungsrichtlinien

In Kapitel 4.2 wurde der Ansatz zur Selbstbeschränkung der delegierten Privilegien durch die Anwender der Grid-Infrastruktur vorgestellt. Kern dieses Ansatzes sind nutzerdefinierte Richtlinien, in denen spezifiziert wird, welche Untermenge der eigenen Privilegien der ausführenden Grid-Ressource zur Verfügung stehen. In diesem Kapitel werden Anforderungen sowie die gewählten Lösungsansätze diskutiert, die eine für den Anwender nutzbare und sichere Implementierung dieses Konzepts (vgl. Kapitel 5) ermöglichen.

4.3.1 Bestandteile und Format

Die Bestandteile nutzerdefinierter Richtlinien ergeben sich aus den Informationen, die für die Entscheidung über einen Ressourcenzugriff (vgl. Kapitel 4.4) benötigt werden. Eine Richtlinie muss dabei Informationen über das Subjekt des Zugriffs enthalten. Im Falle von delegierten Privilegien ist dies in jedem Fall der Nutzer selbst, auch dann, wenn eine Ressource in seinem Namen agiert. Diese Information kann entweder explizit Teil der Richtlinie sein oder sich implizit aus dem Kontext, in dem die Richtlinie durch den PDP der Ressource betrachtet wird, ergeben (vgl. Kapitel 4.3.3). Ist die Richtlinie z. B. digital signiert, geht die Information über das Subjekt aus der Signatur hervor. Weiterer Bestandteil einer Richtlinie ist das Objekt oder Ziel (*target*) des Zugriffs. Die Bezeichner von Objekten variieren dabei im Aufbau je nach Ressourcentyp. Für alle unterstützten Objektarten muss jedoch deren Eindeutigkeit sichergestellt werden. Letzter Bestandteil einer nutzerdefinierten Richtlinie sind Regeln, die für jede mögliche Zugriffsart auf das referenzierte Objekt den Zugriff gestatten oder verweigern.

Zugriffsarten und Adressierung statischer Objekte In Kapitel 4.2.3.1 wurden bereits die Eigenschaften dieser Objektklasse dargelegt, die in Grid-Infrastrukturen von Dateien und Verzeichnissen auf Speicherressourcen repräsentiert wird. Die Verwaltung dieser Objekte, einschließlich der zugehörigen Zugriffsrechte, erfolgt in Infrastrukturen auf Basis der gLite-Middleware durch den *File Catalog* (vgl. Kapitel 2.1.3.2), der auf VO-Ebene betrieben wird.

Die Erstellung von Richtlinien für diese Objekte sollte in der für die Anwender einfachsten Weise ablaufen. Daraus folgt, dass Nutzer in die Lage versetzt werden sollten, die Bezeichner für Objekte zu verwenden, die sie auch für entsprechende Zugriffe nutzen. In der Praxis verwenden Nutzer für diese Art von Objekten einen logischen Bezeichner wie den LFN oder die GUID, der durch den verwaltenden *File Catalog* bei Bedarf in einen physischen Bezeichner wie eine SURL oder TURL übersetzt wird.

Für die Referenzierung mehrerer Objekte in einer Richtlinie ist weiterhin eine möglichst flexible Unterstützung von Wildcards sinnvoll. Es sollte mindestens möglich sein, mehrere Objekte zu referenzieren, deren Bezeichner sich nur in einem Zeichen oder aber ab einem Zeichen bis zum Ende des Bezeichners unterscheidet. Auch eine Referenzierung aller Objekte eines Verzeichnisses sollte mit einem solchen Mechanismus möglich sein.

Zugriffsarten auf Objekte dieser Klasse bestehen aus Lesen (*read*), Schreiben (*write*) sowie Löschen (*delete*). Auch die in einigen Speicherressourcen implementierte Zugriffsart Erstellen (*write-once* oder *create*) sollte unterstützt werden.

Operationen auf und Adressierung von zustandsbehafteten Vorgängen Diese Klasse von Objekten (vgl. Kapitel 4.2.3.2) wird in Grid-Infrastrukturen von Rechenaufträgen und Datentransferoperationen zwischen SE-Diensten repräsentiert. Vorgängen dieser Art werden im Kontext der sie verwaltenden Dienste eindeutige Bezeichner zugeordnet. Diese Bezeichner werden als Ziel (*target*) des Zugriffs Bestandteil der sie betreffenden Richtlinie.

Die möglichen Arten des Zugriffs auf diese Objektklasse wurden bereits in Kapitel 4.2.3.2 vorgestellt. Die Zugriffsart Ausführen (*execute*) muss durch den Mechanismus zur beschränkten Delegation unterstützt werden. Als optional kann hingegen die Unterstützung der Delegation von Berechtigungen für weitere, seltener benötigte Operationen angesehen werden. Mit der Unterstützung dieser Berechtigungen kann es Nutzern beispielsweise ermöglicht werden, das Privileg zum Abbrechen (*cancel*) von Aufträgen separat zu vergeben oder eine solche Operation explizit zu untersagen. Damit wäre eine spezielle Art von Dienstverweigerungsangriffen durch kompromittierte Proxy-Credentials vermeidbar, bei der Angreifer kompromittierte Proxy-Credentials zum Abbruch von Nutzeraufträgen verwenden. Es muss jedoch bezweifelt werden, dass sich für einen Angreifer der Aufwand als

ökonomisch sinnvoll darstellt, erst ein gültiges Proxy-Credential des Nutzers zu erlangen, um nachfolgend mit diesem Aufträge des Nutzers abzubrechen.

Encodierung nutzerdefinierter Richtlinien Die nutzerdefinierten Richtlinien mit den oben genannten Bestandteilen müssen in einem geeigneten Datenformat encodiert werden. Die Eignung dafür ergibt sich aus mehreren Faktoren:

Die Erstellung von Richtlinien direkt durch die Nutzer wie auch indirekt mit Werkzeugunterstützung muss effizient in Bezug auf die dafür benötigte Zeit und mit geringen Kenntnissen durchzuführen sein. Diese Forderungen sollen sicherstellen, dass die Anwender die Mechanismen zur beschränkten Delegation nutzen und die Wahrscheinlichkeit für Fehler in den generierten Richtlinien gering ist.

Die Verarbeitung der Richtlinien durch die PDPs der Ressourcen muss mit geringen Einbußen an Performance des Gesamtsystems erfolgen. Diese Forderung soll sicherstellen, dass die Hürden für den Einsatz in der Praxis durch Verlust an Leistung der Grid-Infrastruktur nicht zu hoch sind.

Das Datenformat muss als offener Standard vorliegen und bereits in Grid-Infrastrukturen (vgl. Kapitel 3.3.1.2) verwendet werden. Beide Forderungen verfolgen das Ziel, Sicherheitslücken durch den Einsatz ungeprüfter Komponenten bzw. Erhöhung der Komplexität durch neue Komponenten zu vermeiden.

Es müssen Werkzeuge verfügbar sein, die eine Unterstützung des Formats in verbreiteten Programmiersprachen bieten. Die Erfüllung dieser Forderung fördert die Verbreitung des Ansatzes durch die Vereinfachung neu zu erstellender Komponenten.

4.3.2 Transport

Der Transport der nutzerdefinierten Richtlinien muss derart erfolgen, dass die PDPs der Grid-Ressourcen Entscheidungen auf ihrer Basis treffen können. Dabei muss für den PDP zu verifizieren sein, dass die Richtlinien durch den Nutzer erstellt und auf dem Weg zur Ressource nicht verändert wurden (vgl. Kapitel 4.3.3).

Sind diese Anforderungen erfüllt, stellt sich die Frage nach der besten Art des Transports der Richtlinien. Prinzipiell lassen sich die Richtlinien mit dem initialen Ressourcenzugriff übertragen oder während der Entscheidung über den Ressourcenzugriff durch den PDP von einem Repository beziehen. In Anlehnung an die in RFC 2904 vorgestellten Kommunikationsmodelle (vgl. Kapitel 2.3.2.2) werden diese Verfahren nachfolgend als Push- bzw. Pull-Verfahren bezeichnet.

Eine dem Agent-Modell äquivalente Implementierung für Grid-Infrastrukturen auf Basis GSI-basierter Middleware würde die Entwicklung eines weiteren Autorisierungsdienstes erfordern, da der Dienstzugriff in diesen Grid-Infrastrukturen nicht

nach dem Agent-Modell abläuft. Dieser müsste dem Ressourcenzugriff vorgeschaltet werden und dessen Entscheidung müssten für die Ressourcen verbindlich sein. Ein Mehrwert oder Gewinn an Sicherheit ist in einer solchen Architektur nicht zu erkennen. Gleichzeitig würde die Performance der Grid-Infrastruktur durch diesen neuen Dienst deutlich beeinflusst sowie ein neuer *Single Point of Failure* geschaffen.

Pull-basierte Übertragung von Richtlinien Die Umsetzung des Pull-Modells erfordert, wie in Abbildung 4.3 dargestellt, die Entwicklung und Einführung eines separaten Dienstes, der als Repository für die nutzerdefinierten Richtlinien fungiert. Dieser Dienst übernimmt die Funktion eines PAP und stellt den PDPs der Ressourcen die Richtlinien der Nutzer zur Verfügung. Soll die gleichzeitige Hinterlegung multipler Richtlinien durch einen Nutzer auf dem Repository ermöglicht werden, z. B. um mehrere Aktionen im Grid mit unterschiedlichen delegierten Privilegien durchführen zu können, ist die Verwendung eindeutiger Referenzen auf die Richtlinien bei Hinterlegung und Dienstzugriff erforderlich. Weiterhin muss ein derartiges Repository permanent zur Verfügung stehen, um die Funktion der Autorisierung anhand nutzerdefinierter Richtlinien sicherzustellen, da es einen *Single Point of Failure* im System darstellt.

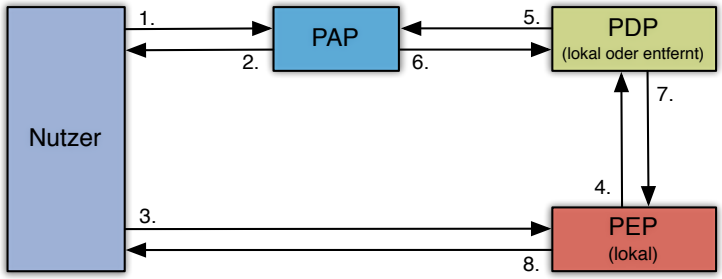


Abbildung 4.3: Transport nutzerdefinierter Richtlinien im Pull-Verfahren

Die Bedeutung der in Abbildung 4.3 dargestellten Abläufe unter Verwendung dieser Architektur erklärt sich wie folgt:

1. Übertragung der nutzerdefinierten Richtlinien auf das Repository
2. Bestätigung über die erfolgreiche Übertragung
3. Zugriff auf die Grid-Ressource und Übermittlung der Referenz auf die nutzerdefinierten Richtlinien
4. Anforderung einer Entscheidung des PDP über Gewährung oder Ablehnung des Zugriffs unter Angabe der Referenz auf die nutzerdefinierten Richtlinien
5. Anforderung der Richtlinien des Nutzers

6. Übermittlung der Richtlinien
7. Mitteilung der Entscheidung über den Ressourcenzugriff
8. Gewährung oder Ablehnung des Dienstes

Push-basierte Übertragung von Richtlinien Bei der Umsetzung des in Abbildung 4.4 dargestellten Push-Modells kann auf zusätzliche Dienste verzichtet werden. In dieser Architektur werden die nutzerdefinierten Richtlinien gemeinsam mit den Dienstzugriffen zu der Ressource übertragen (Schritt 1). Der PEP der Ressource übermittelt die Richtlinien zusammen mit der Anforderung einer Entscheidung über Gewährung oder Ablehnung des Zugriffs an den PDP (Schritt 2). Dieser trifft eine Entscheidung und teilt sie dem PEP mit (Schritt 3). Abschließend setzt der PEP die Entscheidung durch, indem er den Dienst gewährt oder die Verbindung beendet (Schritt 4).

Abbildung 4.4: Transport nutzerdefinierter Richtlinien im Push-Verfahren

Das Push-Modell weist gegenüber dem Pull-Modell deutliche Vorteile hinsichtlich der Komplexität der Architektur auf. Da es keine funktionalen Nachteile gegenüber dem Pull-Modell für diesen Anwendungsfall hat, bildet es die Grundlage des in dieser Arbeit vorgestellten Ansatzes.

4.3.3 Integration in Proxy-Zertifikate

Die Umsetzung des in Kapitel 4.3.2 gewählten Push-Verfahrens für den Transport nutzerdefinierter Richtlinien zu den Grid-Ressourcen ist auf zwei alternativen Wegen möglich. Die erste Alternative ist eine Erweiterung der Protokolle zur gegenseitigen Authentifizierung um eine Sequenz zur Übertragung der nutzerdefinierten Richtlinien. Die Richtlinien sind in diesem Fall durch den Nutzer nach ihrer Erstellung kryptographisch zu signieren, um eine nachträgliche Änderung ihres Inhalts nachweisen zu können. Auch muss sichergestellt werden, dass weitere Ableitungen

des verwendeten Proxy-Zertifikats nicht zu einer Erweiterung der mit diesem delegierten Privilegien führen können. Die Bindung an die nutzerdefinierten Richtlinien ist bei einem solchen Vorgang folglich zu bewahren.

Beide Forderungen, sowohl die einer kryptographischen Signatur als auch die der Bindung nutzerdefinierter Richtlinien an die delegierten Privilegien, führen zur zweiten Alternative für den Transport der nutzerdefinierten Richtlinien. Diese wird durch deren Integration in das an die Grid-Ressource delegierte Proxy-Zertifikat realisiert.

Proxy-Zertifikate verfügen bereits über eine kryptographische Signatur der ausstellenden Entität (*issuer*), die sowohl die Verifizierung der Integrität des Zertifikats als auch seines Ausstellers ermöglicht. Durch die Signatur ist auch die Bindung des Nutzers an die von ihm erstellte und in seinem Proxy-Zertifikat integrierte Richtlinie sichergestellt und kryptographisch überprüfbar.

Die Forderung nach Erhalt der Bindung von nutzerdefinierten Richtlinien an die delegierten Privilegien auch bei weiteren Ableitungen von Proxy-Zertifikaten und Verbreitung dieser auf weiteren Grid-Ressourcen ist mit der Integration in Proxy-Zertifikate ebenfalls erfüllt. Dieser Umstand ergibt sich aus der Tatsache, dass Zertifikate nur dann verifizierbar sind, wenn auch alle vor diesem in der Zertifikatskette stehenden Zertifikate verifiziert werden können. Verifizierende Instanzen erhalten somit auch Kenntnis von nutzerdefinierten Richtlinien in vorhergehenden Elementen der Zertifikatskette und können diese für die Zugriffskontrolle berücksichtigen.

Weitere Vorteile entstehen aus der Option, mehrere Zertifikatserweiterungen mit nutzerdefinierten Richtlinien in eines oder mehrere Proxy-Zertifikate einer Zertifikatskette zu integrieren. Damit sind gezielte Beschränkungen der delegierten Privilegien für unterschiedliche Dienste durch separate Instanzen der nutzerdefinierten Richtlinien möglich. Auch ergibt sich damit die Möglichkeit einer weiteren Einschränkung der delegierten Privilegien auf dem Weg durch die Grid-Infrastruktur.

4.4 Zugriffskontrolle anhand nutzerdefinierter Richtlinien

Die durch den Nutzer erstellten Richtlinien zur Selbstbeschränkung der delegierten Privilegien müssen von entsprechenden Komponenten der Zugriffskontrolle durchgesetzt werden. Dabei ergänzen diese zusätzlichen Komponenten die bestehenden Mechanismen der Zugriffskontrolle, die der Durchsetzung globaler bzw. lokaler Richtlinien dienen.

Neben den Fragen, welche Komponenten des Autorisierungsmodells aus [Org05] benötigt werden, sind auch solche der Effizienz des Ansatzes zu diskutieren. Das Ziel muss dabei sein, die zwangsläufig erzeugten Einbußen an Leistungsfähigkeit der Ressource durch die zusätzlichen Zugriffskontrollmechanismen zu minimieren, ohne deren Wirksamkeit zu beeinträchtigen.

4.4.1 Komponenten der Zugriffskontrollmechanismen

In Abbildung 2.12 auf Seite 32 sind die grundlegenden Komponenten dargestellt, die in einem System zur Zugriffskontrolle eingesetzt werden. Wie der Abbildung zu entnehmen ist, erfolgt der Zugriff auf eine Ressource über deren PEP als erstes Element der Zugriffskontrolle. Dieser fordert für jeden Ressourcenzugriff eine Entscheidung über dessen Gewährung oder Ablehnung bei den für die Ressource zuständigen PDP an und setzt sie nachfolgend durch.

Die Entscheidung über Zugriffe anhand nutzerdefinierter Richtlinien erfordert einen spezialisierten PDP für jede durch diese adressierte Ressourcenart. Für die Zugriffsentscheidung benötigt der PDP neben Informationen über den Kontext des Zugriffs auf die Ressource – bestehend aus handelnder Entität, Zielobjekt und Zugriffsart – die für diesen Ressourcenzugriff gültigen Richtlinien. Diese enthalten durch den Nutzer festgelegte Regeln, die besagen, welche Zugriffe auf welche Objekte gestattet oder untersagt sind.

Durch den Transport der nutzerdefinierten Richtlinien innerhalb des zur Delegation verwendeten Proxy-Zertifikats stehen diese der Ressource und damit dem PEP unmittelbar zur Verfügung. Damit sind zusätzliche PAP unnötig, die der Verwaltung von nutzerdefinierten Richtlinien und deren Transport zum PDP dienen. Es muss jedoch für einen zuverlässigen und gesicherten Transport der Richtlinien vom PEP zum PDP gesorgt werden, falls sich letzterer nicht auf der gleichen Ressource befindet wie der PEP.

Zusätzliche PIP können entfallen, wenn alle Informationen über die benötigten Attribute des Zugriffs direkt durch den PEP an den PDP kommuniziert werden und keine darüber hinausgehenden Informationen benötigt werden. Diese Konstellation ist jedoch nicht die Norm. So werden die Informationen über den Zustand von Aktivitäten des Nutzers (vgl. Kapitel 4.2.3.2) nicht in jedem Fall auf der Ressource selbst verwaltet. In diesem Fall kann es erforderlich sein, den aktuellen Zustand von einem aus Sicht der Ressource entfernten Dienst anzufordern und zur Entscheidung über den Ressourcenzugriff hinzuzuziehen.

4.4.2 Effizienz der Zugriffskontrolle

Neben dem primären Ziel der Zugriffskontrolle, der Verhinderung nichtautorisierter Zugriffe auf die durch diese geschützten Ressourcen, ist eine effiziente Architektur unerlässlich für einen störungsfreien Betrieb. Ziel dieser Forderung ist es, den Einfluss der Zugriffskontrolle auf die Leistungsfähigkeit des Gesamtsystems gering zu halten. Wird diese Randbedingung nicht beachtet, können im Extremfall Dienstverweigerungsangriffe auf das Zugriffskontrollsystem durchgeführt werden, die die ordnungsgemäße Funktion des von diesem geschützten System erheblich beeinträchtigen können.

Zu den zu treffenden Entscheidungen gehört die Platzierung der jeweiligen Komponenten, d. h. ob diese aus Sicht der Ressource lokal oder als entfernter Dienst realisiert werden. Darüber hinaus ist die günstigste Reihenfolge der Entscheidung über Autorisierungsanfragen unter Effizienzgesichtspunkten zu betrachten. Diese Autorisierungsentscheidungen basieren auf den Richtlinien der im Grid relevanten Autoritäten. Sie werden typisch seriell getroffen und nachfolgend logisch UND-verknüpft. Für beide Fragestellungen können sich je nach Ressourcenart unterschiedliche Lösungsansätze als zielführend erweisen.

4.4.2.1 Architektur

Die Entscheidung über die konkrete Ausgestaltung einer Komponente der Zugriffskontrolle ist abhängig von ihrer Art sowie der Fragestellung, ob die zur ordnungsgemäßen Funktion benötigten Informationen auf der Ressource lokal vorliegen oder von einem dritten Dienst bezogen werden müssen. Grundsätzlich ist anzustreben, Zugriffe auf entfernte Dienste zu vermeiden, wenn dadurch die Funktion der Komponente nicht eingeschränkt wird. Zugriffe auf entfernte Dienste belasten die beteiligten Ressourcen erheblich, womit die Leistungsfähigkeit des Gesamtsystems beeinträchtigt wird. Der Grund hierfür ist überwiegend in den verwendeten Authentifizierungsverfahren begründet, die bei jedem Verbindungsaufbau zum Einsatz kommen und deren kryptographische Algorithmen einen erheblichen Rechenaufwand verursachen.

Policy Enforcement Point Der PEP sollte lokal auf der Ressource realisiert werden. Er wird unmittelbar mit dem Dienstzugriff aufgerufen und ist eng mit dem entsprechenden Dienst gekoppelt. Eine Realisierung als entfernter Dienst wäre nur dann effizient, wenn der Dienstzugriff über das Authentifizierungs- und Autorisierungssystem erfolgen würden, wie dies im Agent-Verfahren vorgesehen ist (vgl. Kapitel 2.3.2.2). Da in den betrachteten Infrastrukturen die zur Autorisierung benötigten Informationen jedoch im Push-Verfahren transportiert werden, ist dieser Fall in dieser Arbeit nicht relevant.

Policy Decision Point Der PDP kann lokal, aber auch als entfernter Dienst implementiert werden. Wie in Kapitel 4.4.1 dargelegt wurde, kann es für die Zugriffsentscheidung notwendig sein, auf Informationen zuzugreifen, die von entfernten PIPs zur Verfügung gestellt werden. Werden durch den PDP Informationen von mehreren PIPs abgerufen, kann die Ressource durch einen entfernten PDP signifikant entlastet werden, da sie die notwendigen Operationen zur Authentifizierung nicht durchführen muss. Auch kann der entfernte PDP in diesen Fällen als Aggregationspunkt für nicht-transiente Informationen dienen, so dass eine aufwändige Replizierung dieser Daten auf mehrere Ressourcen vermieden werden kann. Benötigt der PDP jedoch keine weiteren Informationen oder nur Informationen von einem PIP,

so ist es effizienter, ihn als lokale Komponente zu implementieren. Für die lokale Implementierung spricht weiterhin, dass die nutzerdefinierten Richtlinien und die Informationen über die Art des Zugriffs bereits auf der Ressource vorliegen.

Policy Information Point PIPs dienen der Bereitstellung zusätzlicher Autorisierungsinformationen für den PDP. Im Falle der Zugriffsentscheidung anhand nutzerdefinierter Richtlinien stehen die Richtlinien selbst sowie die Art, das Objekt und das Subjekt des Zugriffs dem PDP bereits zur Verfügung. Sie werden durch den PEP zusammen mit der Anforderung einer Zugriffsentscheidung übermittelt. Zusätzliche Informationen werden jedoch für die Autorisierung von Zugriffen auf zustandsbehaftete Objekte benötigt. Die dafür benötigten PIPs sollten nahe der Quelle dieser Informationen platziert werden, d. h. in der Regel auf den Diensten, die diese Informationen verwalten. Das Ziel dieses Vorgehens ist es, definierte Zugangspunkte zu diesen Diensten für Autorisierungskomponenten zu schaffen. Verfügt der Dienst bereits über Mechanismen, die eine beglaubigte oder zumindest authentifizierte entfernte Abfrage der benötigten Informationen erlauben, kann entsprechend auf zusätzliche PIPs verzichtet werden.

4.4.2.2 Verkettung der Zugriffsentscheidungen

Zugriffsentscheidungen werden in Grid-Infrastrukturen durch spezialisierte PDPs getroffen, die die Richtlinien der Autoritäten durchsetzen. Neben den durch die Virtuellen Organisationen erlassenen Richtlinien, die die Privilegien ihrer Mitglieder regeln, werden auf den Ressourcen auch lokale Richtlinien (*site policies*) durchgesetzt. Sie werden üblicherweise als *Black-* bzw. *Whitelists* realisiert, die bestimmten Nutzern den Zugang zu der Ressource gestatten oder verwehren. Die nutzerdefinierten Richtlinien weichen in ihrer Bedeutung von den beiden erstgenannten in der Art ab, dass sie von Nutzern und nicht von einer Grid-Autorität erstellt werden. Sie müssen jedoch ebenfalls auf allen Ressourcen durchgesetzt werden, auf denen die durch sie adressierten Objekte bearbeitet werden. Weitere Arten von Richtlinien sind denkbar, diese werden beispielsweise von den Heimatorganisationen der Nutzer erstellt [GGPW07].

Unabhängig von der Quelle der Richtlinien muss sich eine einzelne Ablehnung durch einen PDP für die Gesamtentscheidung über eine Zugriffsanfrage durchsetzen. Daraus folgt, dass einzig die UND-Verknüpfung der durch die PDPs getroffenen Entscheidungen in der Praxis relevant ist (vgl. Abbildung 4.5 auf der nächsten Seite). Folglich werden bei einer positiven Entscheidung über einen Zugriffswunsch – diese bilden im Produktivbetrieb den größten Anteil – alle PDPs durchlaufen. Die Leistungsfähigkeit des Gesamtsystems kann in diesem Fall nicht durch die Reihenfolge der Abarbeitung der PDPs optimiert werden.

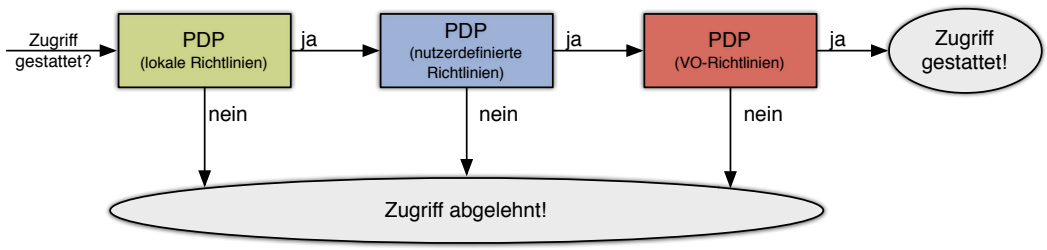


Abbildung 4.5: Verkettung von Policy Decision Points

Im Fall der Ablehnung von Zugriffswünschen ist die Reihenfolge der Abarbeitung der Richtlinien hingegen von Bedeutung, da nun die Kette der PDPs nicht mehr vollständig durchlaufen wird. Somit kann das System bei optimierter Reihung der PDPs effizienter gestaltet werden. Generell ist darauf zu achten, dass PDPs, deren Entscheidungen die höchsten Wahrscheinlichkeiten von negativen Entscheidungen haben und diejenigen, deren Aufwand für die Entscheidungsfindung am geringsten ist, möglichst am Anfang der Kette angeordnet werden.

Der Aufwand, den ein PDP für die Entscheidungsfindung in Form von Belastungen der Systemressourcen sowie in zeitlicher Hinsicht zu betreiben hat, ist dabei von drei Faktoren abhängig:

1. Umfang der zu beachtenden Richtlinien
2. Komplexität des im PDP verwendeten Algorithmus zur Entscheidungsfindung
3. Aufwand durch Abfrage entfernter PIPs oder PAPs

Die Wahrscheinlichkeit der Ablehnung von Zugriffsanfragen durch die eingesetzten PDPs lässt sich lediglich empirisch auf Produktionssystemen ermitteln. Auch können sie je nach System bzw. Infrastruktur, in der die Zugriffskontrollsysteme betrieben werden differieren, so dass die durch den PDP auf die Ressource ausgeübte Belastung bei der Entscheidung über die Position des PDP in der Kette höher gewichtet werden sollte. Letzteres gilt insbesondere für den Fall von Dienstverweigerungsangriffen, bei der der größte Teil der Zugriffswünsche abgelehnt werden muss. Ist das System einer solchen Situation ausgesetzt, kann durch Optimierung der Reihenfolge der PDPs eine Einschränkung der Funktionsfähigkeit des Gesamtsystems vermieden werden. Insbesondere durch *Blacklisting* der angreifenden Entität kann der Einfluss auf das System dabei reduziert werden.

4.5 Unterstützung der Nutzer

Das Ziel bei der Einführung neuer Sicherheitsmechanismen ist die Verbesserung der Sicherheit für Nutzer und Ressourcenbetreiber. Durch die Einführung der nutzerdefinierten Beschränkung delegierter Privilegien soll das Risiko des Missbrauchs kompromittierter Proxy-Credentials gesenkt werden. Dieses Ziel kann nur erreicht werden, wenn die Nutzer die neuen Mechanismen verwenden, da die Formulierung der hierfür erforderlichen Richtlinien in ihrem Verantwortungsbereich erfolgt.

Die Erfahrung zeigt jedoch, dass Sicherheitsmechanismen generell nur dann angewendet werden, wenn ihre Nutzung für die Anwender einfach ist und ihnen im Optimalfall ihre Arbeit erleichtert. Administrative Anordnungen zur Nutzung von Sicherheitsmechanismen sind in der Regel nicht zielführend. Aus diesen Annahmen folgt, dass den Nutzern der Erwerb spezieller Kenntnisse für die Nutzung neuer Sicherheitsmechanismen nicht zugemutet werden kann. Deren Nutzung muss für sie intuitiv erfolgen und sich mit ihren bisherigen Erfahrungen in der Nutzung von IT-Systemen decken.

Die Beschränkung der delegierten Privilegien durch nutzerdefinierte Autorisierungsrichtlinien erfordert die genaue Kenntnis der für die Durchführung der gewünschten Aktionen benötigten Privilegien. Um die Nutzer mit der Erstellung der erforderlichen Richtlinien nicht zu überfordern, sind somit geeignete Mechanismen zu ihrer Unterstützung vorzusehen. Es bieten sich zwei unterschiedliche Ansätze für die verschiedenen Anwendungsbereiche der nutzerdefinierten Richtlinien an (vgl. Kapitel 3.3.1.3). Dies sind zum Einen graphische Benutzeroberflächen, die die Nutzer bei der Erstellung von Richtlinien unterstützen, zum Anderen Mechanismen zu deren automatisierter Erstellung. Letztere können den Vorgang der Erstellung von Richtlinien sowie deren Einbettung in Proxy-Zertifikate für die Nutzer vollständig transparent gestalten.

4.5.1 Automatisierte Erstellung nutzerdefinierter Richtlinien

Der Prozess der Definition und Erstellung von nutzerdefinierten Richtlinien kann für ungeübte und technisch nicht versierte Nutzer eine Hürde mit entsprechendem Fehler- und Frustrationspotential darstellen. Unvollständige, syntaktisch oder semantisch fehlerhafte Richtlinien können zu einem Abbruch von Rechenaufträgen oder zur Verweigerung von Zugriffen auf Daten führen. Idealerweise sollten diese Vorgänge für die Nutzer daher transparent ablaufen, d. h. die Werkzeuge des Nutzerinterfaces sollten sie vollständig von dieser Aufgabe entlasten.

Für die automatisierte Erstellung von nutzerdefinierten Richtlinien eignet sich jedoch nur ein Teil der durch sie adressierbaren Aktionen im Grid. Voraussetzung hierfür ist, dass die durch den Nutzer im Rahmen einer Aktion adressierten Objekte sowie die Arten der Zugriffe auf diese durch das bestehende Nutzerinterface

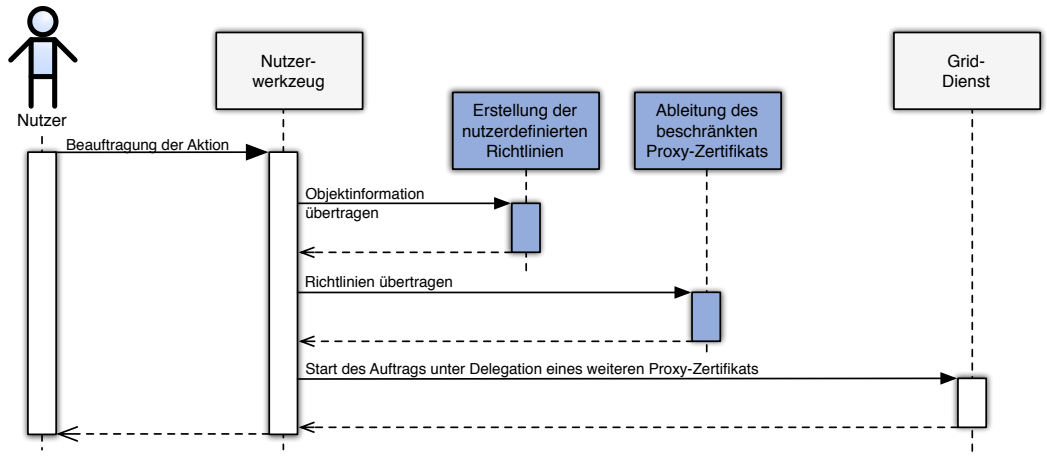


Abbildung 4.6: Ablauf der beschränkten Delegation bei Erweiterung bestehender Nutzerwerkzeuge

implizit aus der durchgeführten Aktion abgeleitet werden können. Nur dann ist die Erstellung der benötigten Richtlinien ohne weiteren Nutzereingriff möglich. Zu dieser Art von Aktionen gehören zustandsbehaftete Vorgänge wie Rechenaufträge oder Aufträge zum Transfer von Daten. Auf Aktionen, die beispielsweise während nichtinteraktiven Rechenaufträgen durchgeführt werden, die für das System folglich bei Beauftragung nicht sichtbar waren, ist diese Art der Erstellung von nutzerdefinierten Richtlinien nicht anwendbar.

Implementierungen dieser Funktionalität können effizient durch eine Erweiterung der bestehenden Nutzerwerkzeuge realisiert werden. Dabei werden die durch den Nutzer für das Auslösen der gewünschten Aktion verwendeten Programme derart verändert, dass bei Feststehen des eindeutigen Objektbezeichners und der gewünschten Zugriffsart der bisherige Programmablauf unterbrochen wird und zusätzliche Funktionen sequentiell aufgerufen werden (in Abbildung 4.6 blau markiert). Dabei ist es unerheblich, ob diese als ergänzende Methoden oder als externe Programme implementiert sind. Diese erstellen die nutzerdefinierte Richtlinie und leiten nachfolgend ein beschränktes Proxy-Zertifikat von dem bereits bestehenden ab. Das beschränkte Proxy-Zertifikat enthält die nutzerdefinierte Richtlinie in einer Zertifikatserweiterung. Beim Aufruf des entfernten Grid-Dienstes wird dieses Proxy-Zertifikat zur Authentifizierung und Autorisierung verwendet. Nachfolgend wird eine Delegation für den Grid-Dienst durchgeführt. Das beschränkte Proxy-Zertifikat ist somit Teil der Zertifikatskette des an den Grid-Dienst delegierten Proxy-Zertifikats, eine nachträgliche Erweiterung der delegierten Privilegien ist somit unmöglich.

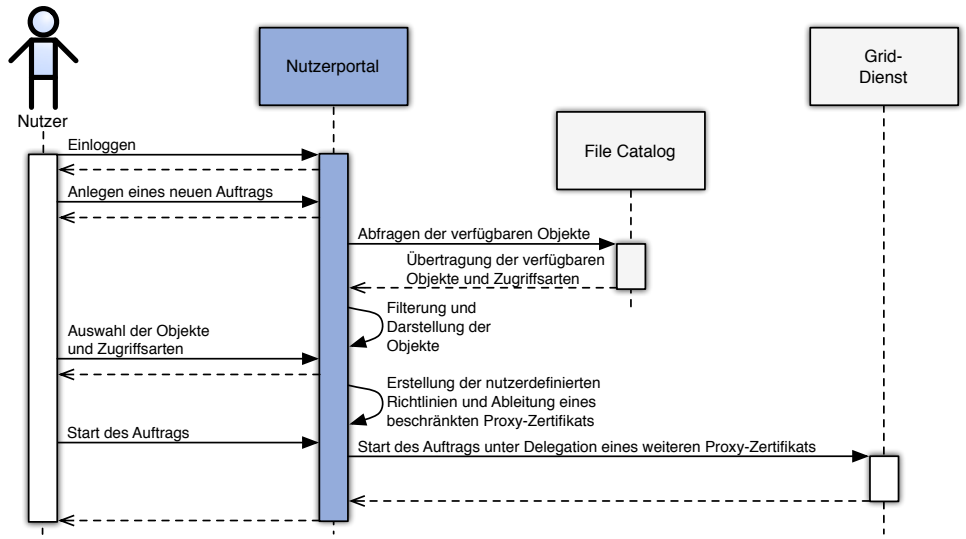


Abbildung 4.7: Ablauf der beschränkten Delegation bei Nutzung einer graphischen Oberfläche

4.5.2 Unterstützung durch graphische Benutzeroberflächen

Die Unterstützung von Nutzern bei der Erstellung von nutzerdefinierten Richtlinien durch graphische Benutzeroberflächen ist in den Fällen erforderlich, in denen Automatismen, wie die in Kapitel 4.5.1 vorgestellten, nicht anwendbar sind. Dies trifft insbesondere auf Objektzugriffe zu, die erst während einer Aktion im Grid, z. B. während eines Rechenauftrags, durchgeführt werden und somit von den Werkzeugen des UI-Systems (vgl. Anhang A.2.1) nicht vorhergesehen werden können. Zugriffe dieser Art werden in aktuellen Grid-Infrastrukturen durch Operationen auf Speichersystemen (vgl. Kapitel 2.1.3.2) repräsentiert, die Unterstützung dieser Operationen muss folglich durch die Implementierung einer derartigen Benutzeroberfläche gegeben sein.

Das funktionale Ziel einer derartigen Implementierung ist das Bereitstellen einer Benutzeroberfläche, die dem Nutzer eine Auswahl von Objekten und Zugriffsrechten auf diese ermöglicht. Das die Oberfläche bereitstellende System muss nachfolgend die automatisierte Erstellung von nutzerdefinierten Richtlinien und die Ableitung von beschränkten Proxy-Zertifikaten durchführen können. Neben diesen auf die beschränkte Delegation fokussierten Funktionen muss die Oberfläche dem Nutzer die für die Erstellung und Beauftragung von Aktionen im Grid benötigten Werkzeuge zur Verfügung stellen. Diese sollten möglichst vollständig nachgebildet werden, um einen Zwang zur Verwendung mehrerer Werkzeuge zu vermeiden sowie dem Nutzer die Umgewöhnung zu erleichtern.

Die Benutzeroberfläche kann dem Nutzer in Form eines lokalen Programms oder eines webbasierten Portals bereitgestellt werden. Die Portallösung hat gegenüber dem lokalen Programm mehrere Vorteile. Sie muss nicht für die durch die Nutzer verwendeten Betriebssysteme angepasst werden, auch ist eine lokale Installation auf den UI-Systemen der Nutzer nicht erforderlich. Da die Nutzer für die Durchführung von Aktionen im Grid über einen Netzzugang verfügen müssen, ist diese Voraussetzung für die Nutzung eines Portals keine Einschränkung.

5 Design und Implementierung

Im Rahmen dieser Arbeit wurde eine Implementierung der in Kapitel 4 vorgestellten Ansätze zur nutzerdefinierten Selbstbeschränkung von Delegationen realisiert. Ziel der Entwicklung war es, einen Nachweis über die Machbarkeit dieser Ansätze zu führen. Die Implementierungsarbeiten wurden durch drei vom Autor am Lehrgebiet Rechnernetze betreute Masterarbeiten [Gro06] [Wil07] [Kun07] unterstützt.

5.1 Grundsätzliche Aspekte der Implementierung

5.1.1 Komponenten

Die Entwicklung wurde am Beispiel der gLite-Middleware durchgeführt. Um eine möglichst vollständige Implementierung des Ansatzes zu erhalten, mussten mehrere Komponenten modifiziert oder entwickelt werden:

- Entwicklung eines auf XACML [Org05] basierenden, erweiterbaren Datenformats für die Formulierung der nutzerdefinierten Richtlinien.
- Erweiterungen der UI-Komponenten um Funktionalitäten zur Generierung der Richtlinien sowie deren Einbettung in Erweiterungen von Proxy-Zertifikaten.
- Entwicklung eines Web-Portals auf Basis von *GridSphere* [NRW04]. Das Portal unterstützt Nutzer bei der Erstellung der nutzerdefinierten Richtlinien für den Zugriff auf SE-Systeme und der Abgabe von Rechenaufträgen an die Grid-Infrastruktur.
- Erweiterung des IO-Service um Komponenten der Zugriffskontrolle. Diese entscheiden anhand nutzerdefinierter Richtlinien über Zugriffsanfragen auf Dateien und Verzeichnisse, die auf SE-Systemen gespeichert sind oder dort abgelegt werden sollen. Die Erweiterung wurde als Beispiel für die nutzerdefinierte Beschränkung der delegierten Privilegien für Zugriffe auf statische Objekte vorgenommen (vgl. Kapitel 4.2.3.1).
- Erweiterung des CE um Komponenten der Zugriffskontrolle. Diese haben zum Ziel, die Ausführung unautorisierter Rechenaufträge mittels kompromittierter Proxy-Credentials zu verhindern. Die Erweiterung wurde als Beispiel für die nutzerdefinierte Beschränkung delegierter Privilegien für Zugriffe auf zustandsbehaftete Objekte vorgenommen (vgl. Kapitel 4.2.3.2).

- Erweiterung des MyProxy-Dienstes mit dem Ziel, die Erneuerung von Delegationen (vgl. Kapitel 3.1.3.2) unter Beibehaltung der Beschränkung von Proxy-Zertifikaten zu unterstützen. Die Erweiterung extrahiert aus zu erneuernden Proxy-Zertifikaten die nutzerdefinierten Richtlinien und bettet diese in die erstellten Proxy-Zertifikate ein.

5.1.2 Verwendete Hard- und Software

Die verwendeten Programmiersprachen für die Implementierung waren weitestgehend durch die zugrunde liegende Grid-Middleware *gLite 3.0* bedingt. Zum Einsatz kamen neben den Programmiersprachen C, C++ und Java die Skriptsprachen Python und Javascript.

Auf den für die Tests der Implementierung genutzten Komponenten des gLite-Testbeds (vgl. Anhang A) war das Betriebssystem *Scientific Linux 3.0* installiert, eine binärkompatible Variante von *Red Hat Enterprise Linux 3*. Dieses basiert auf dem Linux Kernel Version 2.4.21 und stellte zum Zeitpunkt der Entwicklung das einzige durch die gLite-Middleware unterstützte Betriebssystem dar.

Bei Entwicklung und Test des Web-Portals kamen weiterhin der Applikationsserver *Apache Tomcat* [Apa07] in der Version 5.5.20 sowie das *Portal Framework GridSphere* in Version 2.2.7 zum Einsatz.

5.2 Beschränkung der delegierten Privilegien

Wie in Kapitel 4.2 dargelegt, basiert der in dieser Arbeit vorgestellte Ansatz auf einer Selbstbeschränkung der durch die Virtuellen Organisationen zugewiesenen Privilegien durch den Nutzer. Diese Selbstbeschränkung wird in Form von nutzerdefinierten Richtlinien ausgedrückt und durch deren Einbettung in Proxy-Zertifikate an die delegierten Privilegien gebunden.

5.2.1 Encodierung der nutzerdefinierten Richtlinien

Für den Ausdruck der nutzerdefinierten Richtlinien wurde das auf der Auszeichnungssprache XML basierende XACML Schema ausgewählt. Die Syntax von XACML Version 2.0 ist in einem XML-Schema in [Org07a] beschrieben. Darin werden die für ein syntaktisch korrektes Dokument geforderten Sprachelemente in standardisierter Form beschrieben.

XACML erfüllt die in Kapitel 4.3.1 aufgestellten Forderungen. Zudem ist es ein offener Standard und kann somit frei eingesetzt werden. Die erforderliche Werkzeugunterstützung für Syntaxvalidierung und Erstellung von XACML-Dokumenten ist für die verwendeten Programmiersprachen vorhanden.

```
1  ?xml version="1.0" encoding="UTF-8"?> 2
2  <PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
3             xmlns:xacml="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
4             xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
5             xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
                  http://docs.oasis-open.org/xacml/2.0/XACML-CORE/schema_files/
                  access_control-xacml-2.0-policy-schema-os.xsd"
6             PolicyCombiningAlgId="identifier:rule-combining-algorithm:deny-overrides
7             PolicySetId="gLiteUserPolicy">
8   <Target />
9   <Policy ... >
10   ...
11  <Policy ... >
12  </PolicySet>
```

Abbildung 5.1: Struktur eines XACML-Dokuments für den Ausdruck nutzerdefinierter Richtlinien

XACML-Dokumente liegen in Textform vor und können daher prinzipiell auch ohne spezialisierte Werkzeuge mit einem Texteditor bearbeitet werden. Für Nutzer, die über keine Kenntnisse in der Erstellung von Autorisierungsrichtlinien verfügen, ist XACML jedoch zu komplex. Für diese wurden die in Kapitel 5.3 vorgestellten Komponenten zur Nutzerunterstützung entwickelt.

In Abbildung 5.1 ist die Grundstruktur eines XACML-Dokumentes für den Ausdruck von nutzerdefinierten Richtlinien dargestellt. Es handelt sich dabei um ein XML-Dokument in UTF-8 Kodierung (Zeile 1). Nachfolgend beginnt die nutzerdefinierte Richtlinie mit dem *PolicySet*-Element. Die Verwendung dieses Elements bietet den Vorteil, dass in ihm mehrere Richtlinien in Form von *Policy*-Elementen enthalten sein können, die den Zugriff auf verschiedene Objekte regeln.

Die Zeilen 2–5 dienen der Definition des Namensraums sowie zur Angabe des verwendeten XML-Schemas. Damit ist es möglich, den Dokumenttyp zu identifizieren und mittels des Schemas die Syntax des Dokuments zu validieren.

In Zeile 6 wird angegeben, welcher Algorithmus zum Einsatz kommt, wenn mehrere elementare Richtlinien das gleiche Zielobjekt betreffen. Für den *Rule Combining Algorithm* gibt es drei Strategien:

1. *First-Applicable* legt fest, dass die erste Regel Anwendung findet, auf die bezüglich eines Objekts gestoßen wird.
2. *Allow-Overrides* räumt einer den Zugriff gestattenden Regel Vorrang vor ablehnenden ein.
3. *Deny-Overrides* bestimmt, dass eine Regel, die den Zugriff ablehnt, Vorrang vor gestattenden hat.

Zeile 8 enthält ein leeres *Target*-Element. Dieses Element spezifiziert das Subjekt der Richtlinien, d. h. wessen Zugriffsrechte das Dokument beschreibt. Dieses Element ist nach dem XML-Schema obligatorisch, es muss deshalb enthalten sein,

damit das Dokument valide ist. Im Fall der nutzerdefinierten Richtlinien ist diese Information bereits durch die Bindung an ein Proxy-Zertifikat enthalten und muss nicht mehr explizit angegeben werden. Somit enthält dieses Element auch keine weiteren Attribute.

Die Zeilen 9–11 enthalten *Policy*-Elemente. In diesen werden die elementaren Richtlinien spezifiziert, mit denen jeweils der Zugriff auf ein Zielobjekt geregelt wird (vgl. Kapitel 5.2.1.1 und 5.2.1.2). Zeile 12 bildet das Ende der nutzerdefinierten Richtlinie, hier wird das *PolicySet*-Element geschlossen.

5.2.1.1 Elementare Richtlinien für den Datenzugriff

Der Kern der nutzerdefinierten Richtlinien besteht aus einer Anzahl von *Policy*-Elementen. Für die Autorisierung von Zugriffen auf Objekte auf SE-Systemen werden Richtlinien verwendet, deren Attribut *PolicyId* den Wert *FilePolicyX* annimmt (vgl. Abbildung 5.2 auf der nächsten Seite, Zeile 1), wobei „*X*“ die aufsteigende Nummerierung der *FilePolicies* einer nutzerdefinierten Richtlinie darstellt. Die erste *FilePolicy* trägt die Nummer „1“.

Die Spezifikation eines Zielobjekts beginnt am *Target*-Element in Zeile 3 und endet in Zeile 15. XACML erlaubt die Adressierung mehrerer Zielobjekte in einem *Policy*-Element. In den nutzerdefinierten Richtlinien wird diese Funktionalität jedoch nicht unterstützt. Sollen in einer nutzerdefinierten Richtlinie mehrere Zielobjekte adressiert werden, muss für jedes Objekt eine eigene *FilePolicy* erstellt werden.

Adressiert wird das Zielobjekt in Zeile 8, in diesem Fall durch den *Logical File Name* einer Datei. Der in Kapitel 5.4 vorgestellte PDP für den IO-Server unterstützt die Adressierung von Dateien mit dem LFN oder der GUID, da diese die durch den Nutzer direkt verwendeten Adressierungsarten für Dateien in *gLite* sind. Die Zeichenketten, mit denen das Zielobjekt im Zugriffswunsch und in der *FilePolicy* adressiert werden, müssen identisch sein (Zeile 6, *function:string-equal*). Neben der eindeutigen Adressierung eines Zielobjektes können auch Stellvertretersymbole (*wildcards*) verwendet werden, um mehrere Dateien oder Verzeichnisse zu spezifizieren. Es werden die Zeichen „$*$“ für eine Zeichenkette beliebiger Länge sowie „$“ für ein einzelnes Zeichen unterstützt.

In den Zeilen 16–39 sind die Zugriffsarten (vgl. Kapitel 4.2.3.1) aufgeführt, die für das Zielobjekt gestattet (*permit*) oder verboten (*deny*) werden. Damit eine Richtlinie eindeutig ist, müssen alle vier Zugriffsarten aufgeführt sein, anderenfalls ist der Zugriff auf das Zielobjekt zu verweigern.

```
1   <Policy PolicyId="FilePolicy1"
2           RuleCombiningAlgId="identifier:rule-combining-algorithm:deny-overrides">
3    <Target>
4     <Resources>
5      <Resource>
6       <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
7        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
8          lfn:///path/to/my/file
9        </AttributeValue>
10       <ResourceAttributeDesignator AttributeId="
             urn:oasis:names:tc:xacml:1.0:resource:resource-id"
11                                   DataType="http://www.w3.org/2001/XMLSchema#string" /
                                         >
12      </ResourceMatch>
13     </Resource>
14    </Resources>
15   </Target>
16   <Rule Effect="Permit" RuleId="Read">
17    <Target>
18     <Actions>
19      <Action>
20       <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
21        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
22          Read
23        </AttributeValue>
24        <ActionAttributeDesignator AttributeId="
              urn:oasis:names:tc:xacml:1.0:action:action-id"
25                                 DataType="http://www.w3.org/2001/XMLSchema#string" />
26       </ActionMatch>
27      </Action>
28     </Actions>
29    </Target>
30   </Rule>
31   <Rule Effect="Deny" RuleId="Write">
32    ...
33   </Rule>
34   <Rule Effect="Deny" RuleId="Write-Once">
35    ...
36   </Rule>
37   <Rule Effect="Deny" RuleId="Delete">
38    ...
39   </Rule>
40  </Policy>
```

Abbildung 5.2: Beispiel einer elementaren Richtlinie für den Datenzugriff

5.2.1.2 Richtlinien für die Autorisierung von Rechenaufträgen

Das *Policy*-Element für die Autorisierung von Rechenaufträgen (vgl. Abbildung 5.3 auf der nächsten Seite) ist in seiner Grundstruktur identisch mit dem für die Autorisierung von Datenzugriffen. Inhaltlich unterscheidet es sich zum Einen durch eine andere *PolicyId*. Diese hat den Wert *ExecutionPolicyX*, auch hier steht das „*X*“ für eine aufsteigende Nummerierung der Richtlinien mit dem Startwert „1“. Zum Anderen wird das Zielobjekt zwar ebenfalls durch eine Zeichenkette repräsentiert, deren Inhalt ist jedoch eine eindeutige *jobID* (Zeile 8, vgl. Kapitel 3.1.2.2). Der dritte Unterschied zu den *FilePolicies* besteht in den möglichen Zugriffsarten. Die *ExecutionPolicy* kennt lediglich die Zugriffsart *execution*, d. h. das Privileg zur Ausführung eines Rechenauftrags. Auch existiert nur die Möglichkeit, die Zugriffsart zu gestatten (*permit*). Ein Verbot wäre nicht sinnvoll, da die *jobID* erst durch den Nutzer selbst registriert wird und vorher nicht existiert.

```
1   <Policy PolicyId="ExecutionPolicy1"
2           RuleCombiningAlgId="identifier:rule-combining-algorithm:deny-overrides">
3    <Target>
4     <Resources>
5      <Resource>
6       <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
7        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
8         https://lb1.gridlab.uni-hannover.de: 9000/NovxeIp-x-gfOLV40rz7hQ
9        </AttributeValue>
10       <ResourceAttributeDesignator AttributeId="
             urn:oasis:names:tc:xacml:1.0:resource:resource-id"
11                                   DataType="http://www.w3.org/2001/XMLSchema#string" /
                                         >
12      </ResourceMatch>
13     </Resource>
14    </Resources>
15   </Target>
16    <Rule Effect="Permit" RuleId="Execution">
17   <Target>
18    <Actions>
19     <Action>
20      <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function: string-equal">
21       <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
22         execution
23       </AttributeValue>
24       <ActionAttributeDesignator AttributeId="
             urn:oasis:names:tc:xacml:1.0:action:action-id"
25                                  DataType="http://www.w3.org/2001/XMLSchema#string" />
26      </ActionMatch>
27     </Action>
28    </Actions>
29   </Target>
30    </Rule>
31  </Policy>
```

Abbildung 5.3: Beispiel einer elementaren Richtlinie für die Autorisierung eines Rechenauftrags

5.2.2 Integration der nutzerdefinierten Richtlinien in Proxy-Zertifikate

Die nutzerdefinierten Richtlinien werden – wie in Kapitel 4.3.3 dargelegt – in Zertifikatserweiterungen von Proxy-Zertifikaten integriert, um eine Bindung der Richtlinien an den DN des Nutzers herzustellen.

Die Unterscheidbarkeit der Zertifikatserweiterungen ist durch deren eindeutigen *Object Identifier* [WCHK97] (vgl. Kapitel 2.2.2.2) gewährleistet. Die Zertifikatserweiterungen der nutzerdefinierten Richtlinien werden mit OIDs aus dem Namensraum des RRZN mit der Basis ID „1.3.6.1.4.1.18141“ gekennzeichnet, die um den Suffix „3.100.1.1“ bzw. „3.100.2.1“ erweitert werden. Die „3“ kennzeichnet die Abteilung, in der die Erweiterung entwickelt wurde, das *Lehrgebiet Rechnernetze*. Die „100“ steht für Anwendungen des Grid-Computing, die nachfolgende Ziffer „1“ bzw. „2“ kennzeichnet eine *FilePolicy* bzw. eine *ExecutionPolicy*. Weitere Dienstarten können bei Erweiterungen des Ansatzes durch die Zuweisung neuer Bezeichner an dieser Stelle unterstützt werden. Die letzte Ziffer der OID gibt die Versionsnummer der nutzerdefinierten Richtlinien an, die erst im Falle einer zukünftigen Überarbeitung des Ansatzes von Bedeutung ist.

Die Zertifikatserweiterungen für die nutzerdefinierten Richtlinien werden in der Voreinstellung der sie erzeugenden Programme nicht als kritisch (*critical*) markiert.

Damit soll die Ablehnung von Authentifizierungsvorgängen bei Diensten vermieden werden, die diese Erweiterungen nicht unterstützen. Die Mechanismen der Zugangskontrolle anhand nutzerdefinierter Richtlinien auf den Ressourcen können jedoch so konfiguriert werden, dass das Vorhandensein einer nutzerdefinierten Richtlinie notwendig, jedoch nicht hinreichend für eine erfolgreiche Autorisierung ist.

5.3 Nutzerseitige Komponenten

Nutzer von Grid-Ressourcen greifen bisher in der Regel über UNIX-Programme auf der Ebene der Kommandozeile (CLI) auf das Grid zu. Zukünftig wird jedoch angestrebt, auch Fachgebieten die Nutzung des Grid zu ermöglichen, deren Nutzer im Umgang mit UNIX-Systemen unerfahren sind. Für diese werden Web-basierte Lösungen entwickelt, die z. B. das Portal Framework *GridSphere* [NRW04] verwenden. Um beide Nutzergruppen in Hinblick auf die Verwendung beschränkter Delegationen zu unterstützen, wurden sowohl die durch die gLite-Middleware bereitgestellten CLI-Programme erweitert, als auch ein Web-Portal entwickelt.

5.3.1 Modifikation des gLite User Interface Systems

Bevor Nutzer in GSI-basierten Grid-Infrastrukturen Aktionen im Grid durchführen können, müssen sie sich ein Proxy-Zertifikat von ihrem EEZ ableiten. Dafür stellt die gLite-Middleware das CLI-Programm *voms-proxy-init* zur Verfügung. Dieses ist auch für die Anforderung der VO-Attribute von einem oder mehreren VOMS-Servern und deren Einbindung in das zu erstellende Proxy-Zertifikat zuständig. Nach erfolgreicher Beendigung des Programms steht das Proxy-Zertifikat mit dem zugehörigen geheimen Schlüssel als Proxy-Credential in einer lokalen Datei auf dem UI zur Verfügung.

Mit dem neuerstellten Proxy-Credential können nachfolgend Grid-Dienste genutzt werden. Delegationen werden dabei beim Zugriff auf den IO-Service mittels der Programme *glite-*∗ für das Anlegen, Abrufen oder Löschen von Dateien erteilt. Weiterhin wird beim Abgeben von Rechenaufträgen eine Delegation an das Workload-Management-System (vgl. Kapitel 2.1.3.1) erteilt. Dieser Zugriff erfolgt über das CLI-Programm *glite-job-submit*. Dieses Programm generiert auch die den Rechenauftrag eindeutig bezeichnende *jobID* und registriert diese beim LB-Dienst (vgl. Kapitel 2.1.3.4).

Generell stehen für die Erstellung beschränkter Proxy-Zertifikate in diesem Umfeld somit zwei Optionen zur Verfügung. Die erste ist eine Erweiterung von *voms-proxy-init*. Dieses Vorgehen hat den Vorteil, dass nur an einer Stelle Modifikationen vorgenommen werden müssten. Für alle nachfolgenden Operationen wird dann die gleiche nutzerdefinierte Richtlinie verwendet. Wird eine andere Beschränkung

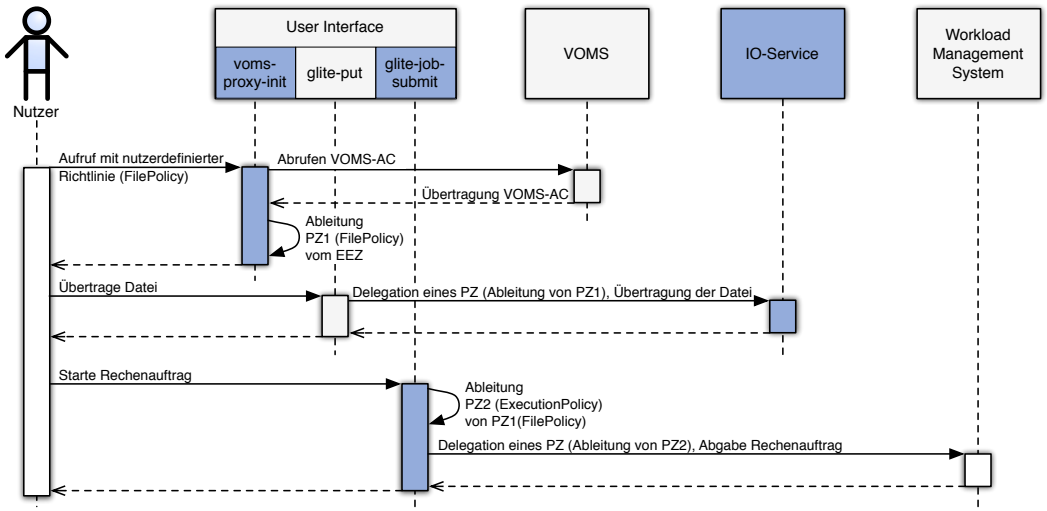


Abbildung 5.4: Vorbereitung und Abgabe eines Rechenauftrags mit den erweiterten Komponenten des User Interface

des Proxy-Zertifikats benötigt, muss durch einen erneuten Aufruf von *voms-proxy-init* eine andere Richtlinie in das neu zu erstellende Proxy-Zertifikat eingebunden werden. Für die Autorisierung eines Rechenauftrags mittels einer *ExecutionPolicy* ist die ausschließliche Erweiterung von *voms-proxy-init* jedoch untauglich, da bei Ausführung von *voms-proxy-init* die *jobID* für den Rechenauftrag noch nicht feststeht.

Die zweite Option ist die Erweiterung der Programme des UI, mittels derer auf den IO-Service zugegriffen bzw. Rechenaufträge an das WMS abgegeben werden, um Mechanismen zur Generierung beschränkter Proxy-Zertifikate. Diese Option bietet den Nutzern ein Höchstmaß an Flexibilität, ist aber in der Bedienung deutlich aufwändiger, da bei jedem Programmaufruf eine in das zu erstellende Proxy-Zertifikat einzubindende Richtlinie angegeben werden muss.

Die gewählte Lösung ist eine Kombination aus beiden Optionen (vgl. Abbildung 5.4, erweiterte Komponenten in blau). Für Rechenaufträge werden sowohl eine *FilePolicy* als auch eine *ExecutionPolicy* benötigt, um die während des Rechenauftrags durchgeführten Zugriffe auf Datendienste beschränken zu können. Daher wurde das Programm *voms-proxy-init* um eine Funktion erweitert, die die durch den Nutzer erstellte und als Textdatei vorliegende *FilePolicy* in eine entsprechende Erweiterung des erstellten Proxy-Zertifikats einbindet. Der Nutzer kann damit nach einmaligem Aufruf von *voms-proxy-init* mehrere Operationen auf dem IO-Service durchführen (in Abbildung 5.4 beispielhaft ein *glite-put*), ohne weitere beschränkte Proxy-Zertifikate erstellen zu müssen. Zusätzlich wurde das als Python-Skript vor-

liegende *glite-job-submit* erweitert. Es leitet von dem durch *voms-proxy-init* erstellten Proxy-Zertifikat ein weiteres ab und integriert in dieses ohne weiteren Nutzereingriff eine Zertifikatserweiterung für eine *ExecutionPolicy*. Diese enthält die selbstgenerierte *jobID* und eine Regel, die die Ausführung des zugehörigen Rechenauftrags gestattet. Während der Abgabe des Rechenauftrags erhält das WMS damit eine Delegation, in deren Zertifikatskette sowohl eine *FilePolicy* als auch eine *ExecutionPolicy* enthalten ist.

5.3.2 Web-Portal zur Nutzerunterstützung bei der Erstellung der Richtlinien

Wie in Kapitel 4.5.2 dargelegt, ist es notwendig, Anwender bei der Erstellung der nutzerdefinierten Richtlinien in den Fällen zu unterstützen, in denen diese nicht automatisch generiert werden können. Eine derartige Funktionalität wurde in Form eines Web-Portals implementiert, das auf dem Portal Framework *GridSphere* und dem *Apache Tomcat* Applikationsserver aufsetzt. Der Zugriff auf die Grid-Ressourcen erfolgt über die Komponenten des *User Interface*, wodurch eine Wiederverwendung der in Kapitel 5.3.1 vorgestellten Komponenten möglich war.

Das Ziel bei der Erstellung des Web-Portals war es, den Nutzer möglichst vollständig von der Notwendigkeit zur Verwendung der CLI-Programme zu entlasten (vgl. Kapitel 4.5.2). Der Nutzer sollte bei Verwendung des Web-Portals möglichst nur noch die schematisch in Abbildung 4.7 auf Seite 88 dargestellten Handlungen ausführen müssen. Dafür waren die folgenden Funktionalitäten zu implementieren bzw. nach dem Vorbild der CLI-Programme nachzubilden:

- Verwaltung der Proxy-Credentials des Nutzers. Der Nutzer sollte die Option haben, entweder ein bestehendes Proxy-Credential auf das Web-Portal zu kopieren oder eines von einem MyProxy-Service zu beziehen.
- Abrufen der VO-Attribute des Nutzers von VOMS-Diensten und zur Generierung von Proxy-Zertifikaten mit eingebetteten VO-Attributen. Dabei sollte der Nutzer die zur Verfügung stehenden VO-Attribute angezeigt bekommen und auswählen können, welche er delegieren möchte.
- Erstellung der Beschreibung eines Rechenauftrags in der *Job Description Language*. Der Nutzer muss dafür das auszuführende Programm sowie die anzuwendenden Argumente spezifizieren können. Weiterhin sollte er die Mindestanforderungen der für den Rechenauftrag benötigten Umgebung sowie die Ein- und Ausgabedateien spezifizieren können.
- Kopieren der benötigten Dateien. Das Web-Portal sollte dem Nutzer die Option bereitstellen, Dateien, wie die während eines Rechenauftrags benötigten Programme oder Eingabedateien, auf das Portal sowie Ergebnisdateien eines Rechenauftrags auf sein lokales Arbeitsplatzsystem zu kopieren.
- Abrufen und Anzeigen der Objekte des *File Catalog* Dienstes. Dabei sollte der Nutzer alle ihm zur Verfügung stehenden Objekte einschließlich der Zugriffs-

rechte angezeigt bekommen. Auch sollte er in die Lage versetzt werden, die gewünschten Objekte und Zugriffsrechte für eine nutzerdefinierte Richtlinie auszuwählen.

- Generierung der nutzerdefinierten Richtlinie aus der Auswahl des Nutzers und Ableitung eines beschränkten Proxy-Zertifikats.
- Abfrage des WMS nach zur Verfügung stehenden CE-Systemen unter Berücksichtigung der spezifizierten Anforderungen. Anzeige der CEs und Auswahlmöglichkeit für den Nutzer, welches CE genutzt werden soll.
- Starten von Rechenaufträgen durch Übergabe an das WMS und Anzeige von deren Status.

Die aufgeführten Funktionalitäten wurden in drei Aufgabenbereiche unterteilt und in Form von *Java Portlets* implementiert. Die *Portlets* sind zustandsbehaftet, der Nutzer kann folglich zwischen ihnen hin- und herwechseln oder sich ab- und wieder anmelden, ohne dass seine Eingaben verloren gehen. Die Funktionalitäten der *Portlets* wurden wie folgt gruppiert (für eine ausführliche Darstellung der Abläufe vgl. Anhang B):

- Das *Credential Portlet* ist zuständig für die Verwaltung von Proxy-Credentials der Nutzer. Es stellt die Funktionalität zum Abruf der VO-Attribute von VOMS-Diensten bereit und bietet die VO-Attribute zur Auswahl an.
- Das *Resources Portlet* organisiert die zur Verfügung stehenden Grid-Ressourcen und stellt sie dem Nutzer dar. In einer späteren Version kann dieses *Portlet* auch dafür genutzt werden, dem Nutzer die Option zu geben, selbstständig weitere Grid-Ressourcen dem Portal hinzuzufügen.
- Das *Job Submission Portlet* stellt alle Funktionalitäten bereit, die der Erstellung von Rechenaufträgen und den nutzerdefinierten Richtlinien dienen. Um erfahrenen Nutzern eine Kontrollmöglichkeit zu geben, ob die erstellten Dateien zur Beschreibung von Aufträgen bzw. die erstellten Richtlinien ihren Intentionen entsprechen, werden diese jeweils in Textform angezeigt. Das *Portlet* dient weiterhin der Kommunikation mit dem WMS und dem LB-Service zum Start von Rechenaufträgen sowie der Anzeige ihres Zustands.

Die Kommunikation der *Portlets* mit den Komponenten des UI erfolgt über eine gemeinsame Wrapper-Klasse. Diese nimmt Anfragen der *Portlets* entgegen und führt die entsprechenden CLI-Programme aus. Die nachfolgenden Ausgaben werden gefiltert und an das *Portlet* übertragen.

5.4 Erweiterung der Autorisierungsfunktionalität des gLite IO-Service

Die Dateien der Nutzer werden in Grid-Infrastrukturen auf Basis der gLite-Middleware auf SE-Systemen mit SRM-Schnittstelle gespeichert. Der Zugriff auf diese Daten durch die Nutzer erfolgt jedoch über einen vorgeschalteten IO-Service (vgl. Kapitel 3.1.3.1). Allein dieser verfügt über die Berechtigung, direkt auf die SE-Systeme zuzugreifen. Diese Architektur wurde gewählt, da die gängigen SRM-Implementierungen *dCache* [dCa07] und *Castor* [CER07] zum Zeitpunkt der Entwicklung der gLite-Middleware nur über rudimentäre Mechanismen der Zugriffskontrolle verfügten.

Der IO-Service ist für die Durchsetzung der in den *File Catalog* Diensten jedem Objekt zugeordneten Zugriffsrechte bzw. ACL zuständig. Diese Zugriffsrechte werden durch die den *File Catalog* betreibenden Virtuellen Organisationen oder durch den Besitzer eines Objekts gesetzt. Entscheidungen über Objektzugriffe werden durch den im *File Catalog* implementierten *File Authorization Service* (FAS) gefällt. Der FAS agiert folglich aus Sicht des IO-Service, der lediglich die Funktion eines PEP wahrnimmt, als entfernter PDP.

Für die Implementierung der Zugriffskontrolle anhand der nutzerdefinierten Richtlinien kamen prinzipiell alle drei Komponenten der Datendienste in Frage, IO-Service, *File Catalog* sowie das SE. Die Erweiterung des *Storage Element* unter Beibehaltung der Architektur und der verwendeten Protokolle erwies sich aus zwei Gründen als nicht möglich:

1. Die nutzerdefinierten Richtlinien stehen lokal auf dem IO-Service und dem *File Catalog* zur Verfügung, da die Authentifizierung an beiden mittels des beschränkten Proxy-Zertifikats erfolgt. Die Authentifizierung beim Zugriff auf das SE erfolgt hingegen mit dem Host-Zertifikat des IO-Service. Die nutzerdefinierten Richtlinien sind folglich nicht verfügbar und müssten durch eine Erweiterung des SRM-Protokolls dem SE zugänglich gemacht werden.
2. Auf dem IO-Service und dem *File Catalog* stehen alle benötigten Informationen über den ausgeführten Zugriff zur Verfügung. Dazu gehören das Zielobjekt in Form eines LFN oder einer GUID, die gewünschte Zugriffsart sowie das handelnde Subjekt. Für das SE ist das handelnde Subjekt der IO-Service und das Zielobjekt ist durch eine SURL ausgedrückt. Letztere könnte durch eine Abfrage des *File Catalog* wieder in den in den Richtlinien referenzierten LFN zurückübersetzt werden. Das ursprünglich handelnde Subjekt ist jedoch nur dem IO-Service und dem *File Catalog* bekannt.

Die Entscheidung für die Implementierung der Zugriffskontrolle anhand der nutzerdefinierten Richtlinien fiel aus Effizienzgründen (vgl. Kapitel 4.4.2) zugunsten einer Erweiterung des IO-Service. Auf diesem kann die Entscheidung unmittelbar nach dem Zugriff auf Basis lokaler Informationen gefällt werden. Bei einer negativen Zugriffsentscheidung auf Basis der nutzerdefinierten Richtlinien kann auf eine Kon-

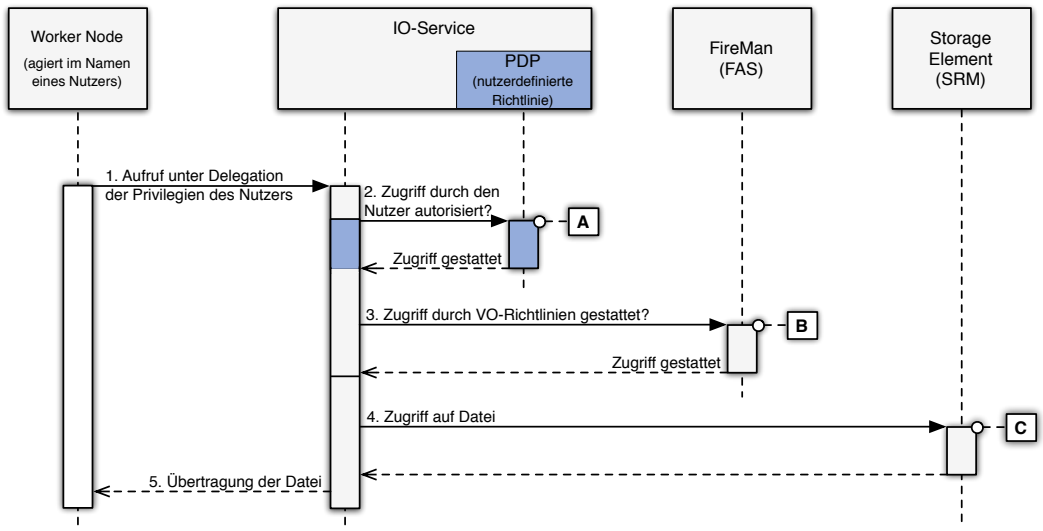


Abbildung 5.5: Lesender Datenzugriff (modifizierte Zugriffskontrolle)

taktierung des FAS zur Entscheidung über den Zugriff anhand dessen Richtlinien verzichtet werden. Eine Implementierung auf dem *File Catalog* hingegen hätte bei jedem Zugriff dessen Kontaktierung durch den IO-Service nach sich gezogen.

5.4.1 Abläufe bei der Zugriffskontrolle eines Datenzugriffs

In Abbildung 5.5 ist ein exemplarischer Datenzugriff dargestellt. Es wird dabei vorausgesetzt, dass das Zielobjekt des Zugriffs auf dem SE vorhanden und im *File Catalog* eingetragen ist. Der Zugriff auf das Zielobjekt soll lesend erfolgen, der Nutzer verfügt über die dafür benötigten Privilegien. Weiterhin soll der *Worker Node* über eine gültige Delegation verfügen. Das dafür erstellte Proxy-Credential wurde mittels einer nutzerdefinierten Richtlinie in den delegierten Privilegien beschränkt. In der Richtlinie enthalten ist eine elementare *FilePolicy* mit einer Regel bezüglich des Zielobjekts. Der Zugriff läuft wie folgt ab:

1. Der WN kontaktiert den IO-Service, dabei wird eine gegenseitige Authentifizierung (vgl. Kapitel 2.2.3) durchgeführt. Der WN verwendet dafür das durch den Nutzer delegierte Proxy-Zertifikat und den zugehörigen geheimen Schlüssel. Bei diesem Vorgang extrahiert der PEP des IO-Service die enthaltene nutzerdefinierte Richtlinie.
2. Der PEP fordert eine Autorisierungsentscheidung beim PDP an (**A**). Diese wird lokal anhand der nutzerdefinierten Richtlinie getroffen. Sollte der Zugriff durch den PDP untersagt werden, würde der Vorgang an dieser Stelle durch

Übermittlung einer entsprechenden Fehlermeldung abgebrochen. In diesem Fall fällt die Entscheidung positiv aus.

3. Der PEP fordert nun eine Autorisierungsentscheidung vom entfernten PDP des *File Catalog*, dem FAS, an (**B**). Dieser fällt seine Entscheidung anhand der VO-Richtlinien. Fällt auch diese Entscheidung positiv aus, ist der Zugriff aus Sicht des IO-Service gestattet.
4. Der IO-Service versucht nun, die Datei vom SE zu beziehen. Für die gegenseitige Authentifizierung verwendet er sein Host-Zertifikat. Nach erfolgreicher Authentifizierung entscheidet das SE (**C**), ob dem IO-Service der Zugriff gestattet ist. Im positiven Fall überträgt er nachfolgend die Datei an den IO-Service.
5. Die Datei wird an den *Worker Node* übertragen.

5.4.2 Implementierung des Policy Decision Point

Das Programm *IODaemon* bildet den Hauptbestandteil des IO-Service und implementiert auch den PEP für Entscheidungen über Zugriffe auf Datenobjekte. Da nach der Authentifizierung der zugreifenden Entität am *IODaemon* dort auch das zur Authentifizierung verwendete Proxy-Zertifikat einschließlich der gesamten Zertifikatskette vorliegt, wurde der *IODaemon* um die benötigten Funktionalitäten erweitert. Dies schließt die Extraktion der nutzerdefinierten Richtlinien aus den Proxy-Zertifikaten sowie die Entscheidung über die und die Durchsetzung der darin ausgedrückten Richtlinien ein. Der erforderliche PDP wurde folglich direkt in den *IODaemon* integriert. Dazu wurden die Methoden *getPolicyExtensionData()* und *parseUserPolicy()* entwickelt. Die Methode *getPolicyExtensionData()* extrahiert anhand der OID die nutzerdefinierten Richtlinien aus den Zertifikaten der Zertifikatskette, die *FilePolicy*-Elemente enthalten und liefert eine Datenstruktur vom Typ *XMLByte* zurück. Die Methode *parseUserPolicy()* durchläuft diese Datenstruktur und fügt die Richtlinien einem *myPolicies* Objekt hinzu.

Für die Evaluierung der Richtlinien wurde der IO-Service um zwei Klassen erweitert. Die Klasse *Policies* nimmt die einzelnen Richtlinien auf und verfügt über Methoden, neue Richtlinien dem Vektor *policies* hinzuzufügen (*addPolicy()*) und diese Richtlinien anhand eines konkreten Zugriffs auszuwerten (*evaluatePolicy()*). Der Vektor *policies* enthält Instanzen der zweiten Klasse (*Policy*). Jede dieser Instanzen enthält genau eine Regel bezüglich eines Zielobjekts, d. h. für jede *FilePolicy* wird eine neue Instanz der Klasse *Policy* angelegt. Diese enthält den Pfad zum Zielobjekt in LFN oder GUID Notierung sowie bool'sche Variablen bezüglich der Zugriffsrechte auf dieses Objekt. Darüber hinaus verfügt die Klasse über Methoden, die den Pfad des Zielobjekts ausgeben (*getPath()*) und Auskunft darüber erteilen, ob ein bestimmter Zugriff auf das Zielobjekt laut Regel gestattet oder untersagt ist (*get∗()*).

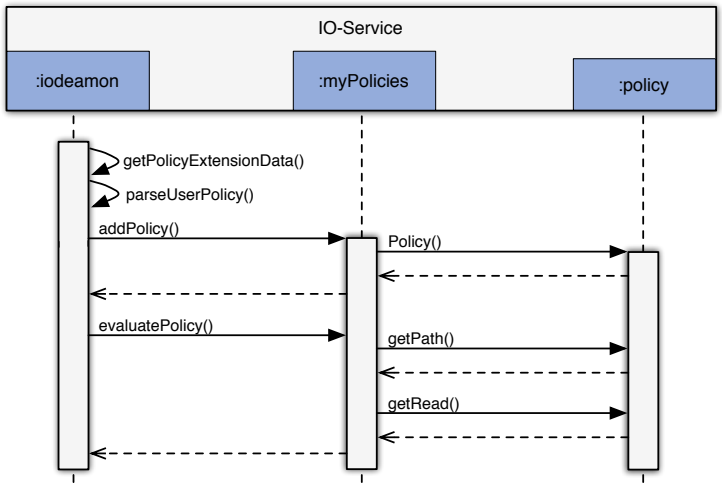


Abbildung 5.6: Sequenzdiagramm der modifizierten Zugriffskontrolle im IO-Service

In Abbildung 5.6 ist der interne Ablauf im *IODaemon* während einer Zugriffskontrolle anhand von nutzerdefinierten Richtlinien verdeutlicht. Die Abbildung stellt die internen Abläufe von Schritt *A* der Zugriffskontrolle in Abbildung 5.5 auf Seite 102 dar:

1. Nach einem Zugriffswunsch werden die nutzerdefinierten Richtlinien durch die Methode *getPolicyExtensionData()* aus der Zertifikatskette extrahiert.
2. Aus der enthaltenen XACML-Datenstruktur wird durch *parseUserPolicy* ein *myPolicies* Objekt aufgebaut.
3. Für jede enthaltene *FilePolicy* wird ein *policy* Objekt mit dem referenzierten Zielobjekt und den gestatteten oder untersagten Zugriffsarten initialisiert.
4. Die *evaluatePolicy()* Methode wird mit dem aktuellen Zugriffswunsch aus Zielobjekt und Zugriffsart aufgerufen.
5. Die vorhandenen Richtlinien werden nach dem Zielobjekt und der Zugriffsart des aktuellen Zugriffs durchsucht. Bei Vorhandensein mehrerer Richtlinien mit dem gleichen Zielobjekt wird nach der *Deny-Overrides* Strategie über den Zugriff entschieden (vgl. Kapitel 5.2.1).

5.5 Erweiterung der Autorisierungsfunktionalität des gLite-CE

Rechenaufträge werden in gLite-Infrastrukturen durch die Komponenten *Workload-Management-System*, *Computing Element* und *Worker Node* bearbeitet. Die Verwaltung der Rechenaufträge sowie ihrer zugeordneten Zustände wird durch das *Logging & Bookkeeping* anhand der eindeutigen *jobID* wahrgenommen (vgl. Kapitel 3.1.1).

5.5.1 Zugriffskontrolle anhand des Zustands von Aufträgen

Die *jobID* wird durch den Nutzer vor Abgabe des Rechenauftrags an das WMS beim LB-Service registriert. Um die delegierten Berechtigungen auf die Bearbeitung genau dieses Rechenauftrags zu beschränken, wird ein beschränktes Proxy-Credential zu dessen Beauftragung verwendet. Dieses enthält eine *ExecutionPolicy* mit einer Regel, die die Ausführung des Auftrags mit der registrierten *jobID* gestattet (vgl. Kapitel 5.2.1.2). Während der Bearbeitung des Auftrags werden durch die beteiligten Komponenten unter der *jobID* Zustandsmeldungen abgelegt (vgl. Abbildung 3.1 auf Seite 44). Dafür wird die *jobID* eines Auftrags der jeweiligen Komponente durch ihren Vorgänger in der Kette übermittelt.

Der naheliegende Ansatz zur Zugriffskontrolle anhand der *jobID* wäre nun, ausschließlich einen einfachen Vergleich der durch die Middleware auf die beteiligten Komponenten übertragenen *jobID* mit der in der *ExecutionPolicy* des delegierten Proxy-Zertifikats referenzierten vorzunehmen. Bei Übereinstimmung der beiden würde der Rechenauftrag zur Bearbeitung akzeptiert. Dieser Ansatz ist jedoch nicht ausreichend sicher. Es besteht die Möglichkeit, dass ein Angreifer, der erfolgreich ein beschränktes Proxy-Credential in seinen Besitz gebracht hat, direkt einen Rechenauftrag an ein CE abgibt. Dabei gibt der Angreifer auch die *jobID* an, unter der der Auftrag verwaltet werden soll, sodass ein Zugriffskontrollmechanismus auf dem CE davon ausgehen würde, dass der Rechenauftrag rechtmäßig ist.

Aus diesem Grund wurde ein Zugriffskontrollmechanismus entwickelt, der den Zustand des Rechenauftrags mit in die Entscheidung über dessen Zulassung oder Ablehnung einbezieht. Dafür wurden die möglichen Zustände von Rechenaufträgen analysiert und diejenigen Zustände festgestellt, die bei Eintreffen eines Rechenauftrags am Gatekeeper des CE (vgl. Kapitel 3.1.2.3) für einen noch unbearbeiteten Rechenauftrag valide sind. Die validen Zustände sind „Submitted“, „Waiting“ sowie „Ready“ (blau eingefärbt in Abbildung 5.7 auf der nächsten Seite). In allen anderen Zuständen kann sich ein Rechenauftrag nur befinden, nachdem er bereits einmal durch ein *Computing Element* bzw. einen *Worker Node* bearbeitet wurde. In diesen Fällen muss folglich davon ausgegangen werden, dass ein Angreifer versucht, mit einem kompromittierten, beschränkten Proxy-Credential einen Rechenauftrag unter einer wiederverwendeten *jobID* abzugeben. Die Bearbeitung des Auftrags muss folglich abgelehnt werden.

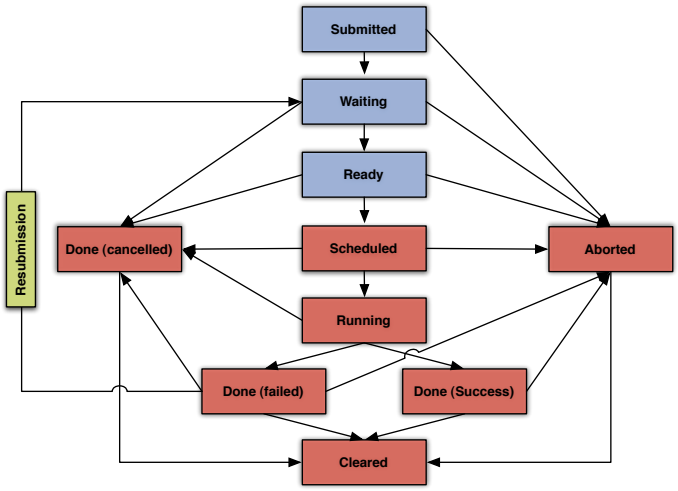


Abbildung 5.7: Zustände eines Rechenauftrags in gLite-Infrastrukturen

Eine Sonderstellung nimmt der Zustand „Resubmission“ ein. Schlägt die Ausführung eines Rechenauftrags in gLite-Infrastrukturen fehl, versucht die Middleware bis zu dreimal, diesen Auftrag neu zu starten. Dabei wird der wiederholte Ausführungsversuch durch einen derartigen Zustand im LB-Service markiert. Soll diese Funktionalität durch den Zugriffskontrollmechanismus unterstützt werden, ist die Abfolge der Zustände „Done(Failed)→Resubmission→Waiting→Ready“ ebenfalls als valide anzuerkennen. Aufgrund der Schwierigkeit, zweifelsfrei über die Rechtmäßigkeit einer *Resubmission* zu entscheiden, sollte diese Funktionalität bei hohen Sicherheitsanforderungen nicht unterstützt werden.

Der entwickelte Zugriffskontrollmechanismus basiert auf zwei Annahmen, die für ein sicheres Funktionieren von essentieller Bedeutung sind und beide durch den in gLite-Infrastrukturen genutzten LB-Service erfüllt werden:

1. Die Zustandsmeldungen dürfen nur nach starker Authentifizierung entgegengenommen werden. Auch darf nur der Nutzer selbst bzw. eine in seinem Namen handelnde Entität für die Abgabe einer derartigen Meldung zu seinen Rechenaufträgen autorisiert werden. Andernfalls wäre es möglich, die Zugriffskontrolle durch Abgabe von Meldungen derart zu manipulieren, dass Rechenaufträge fälschlicherweise abgewiesen würden.
2. Es dürfen ausschließlich Zustandsmeldungen auf dem verwaltenden Dienst hinzugefügt oder gelesen werden. Eine Löschung von Einträgen muss unmöglich sein. Andernfalls wäre eine Manipulation der Zugriffskontrolle durch Löschen der Zustandsmeldungen und somit eine erfolgreiche erneute Abgabe von Rechenaufträgen unter derselben *jobID* möglich.

5.5.2 Auswahl des CE für die Erweiterung

Für die Durchsetzung von nutzerdefinierten Richtlinien auf den für die Ausführung von Rechenaufträgen zuständigen Komponenten wurde das *Computing Element* ausgewählt. Gegen die Implementierung des benötigten PEP auf dem WMS sprach die Tatsache, dass man Rechenaufträge auch direkt auf ein CE abgeben kann, ohne einen WMS zu verwenden. Dadurch könnten die Zugriffskontrollmechanismen auf dem WMS umgangen werden. Die Implementierung der Funktionalität auf dem *Worker Node* wäre im Ablauf der Bearbeitung von Rechenaufträgen zu spät gewesen, da diese, wenn sie auf WN eintreffen, bereits gestartet wurden. Für die Implementierung auf dem CE sprach die Tatsache, dass dieses bereits die Zugriffskontrollen für Rechenaufträge anhand lokaler und VO-Richtlinien vornimmt. Weiterhin ist es die letzte Grid-Ressource vor den *Worker Nodes*, eine Umgehung durch direkte Abgabe von Rechenaufträgen ist nicht möglich.

Für die Implementierung des PDP für die nutzerdefinierten Richtlinien standen zwei Varianten zur Auswahl. Die erste war die Implementierung eines aus Sicht des CE entfernten PDP auf dem LB-Service. Dieser würde für die Entscheidung auf die lokalen Daten aus der Datenbank des LB-Service zugreifen. Dessen Aufgabe entspräche somit der eines PIP. Als Protokoll für die Anforderung von Entscheidungen durch den PEP vom entfernten PDP wäre das *XACML-Request/Response*-Protokoll einsetzbar. Die zweite Option war die Implementierung sowohl des PEP als auch des PDP auf dem CE. Der LB-Service wäre als entfernter PIP in die Entscheidungsfindung eingebunden, um Informationen über den Status des betreffenden Rechenauftrags zu erhalten. Die entsprechenden Mechanismen und Protokolle zur Abfrage des LB-Service existieren bereits in der API der gLite-Middleware.

Beide Optionen waren sowohl funktional als auch bezüglich der zu erwartenden Belastung der beteiligten Systeme äquivalent. Die Entscheidung fiel daher zugunsten der zweiten Option, da diese mit geringerem Aufwand und somit weniger zu erwartenden Fehlern implementiert werden konnte (vgl. Kapitel 5.5.3).

5.5.3 Implementierung des PDP

Die Durchsetzung von Zugriffsentscheidungen wird im gLite-CE durch den *Gatekeeper* vorgenommen (vgl. Kapitel 3.1.2.3 und Abbildung 3.1). Dieser fordert Zugriffsentscheidungen beim *Local Centre Authorization Service* (LCAS) an, der somit die Rolle eines PDP für das CE einnimmt. Intern ist LCAS modular aufgebaut und enthält mehrere PDPs, die Zugriffsentscheidungen anhand lokaler und VO-Richtlinien fällen. Mittels eines Plugin-Mechanismus ist LCAS erweiterbar ausgeführt.

Die Implementierung der benötigten Funktionalitäten konnte somit in Form eines als Plugin ausgeführten PDP vorgenommen werden. Das Plugin ist in *C* als monoli-

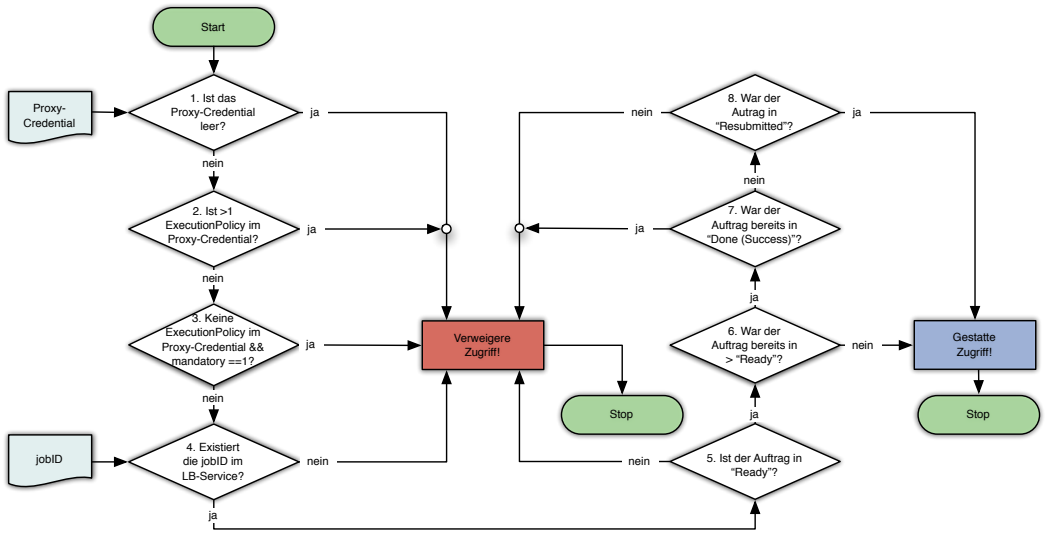


Abbildung 5.8: Ablauf der Zugriffskontrolle anhand nutzerdefinierter Richtlinien

thisches Programm ausgeführt. Es greift über die gLite-API auf den LB-Service zu, um den Status des in der nutzerdefinierten Richtlinie referenzierten Rechenauftrags zu erhalten. Ein separater PEP wurde nicht benötigt.

Der Ablauf der Zugriffsentscheidung ist wie folgt (vgl. Abbildung 5.8):

1. Nach der Initialisierung des Plugin wird das Proxy-Credential eingelesen, das bei Erteilen des Auftrags delegiert wurde. Ist es leer oder fehlerhaft, wird der Zugriff verweigert. Andernfalls wird der nächste Schritt ausgeführt.
2. Ist ein gültiges Proxy-Credential vorhanden, dessen Proxy-Zertifikate jedoch mehr als eine nutzerdefinierte Richtlinie mit *ExecutionPolicy*-Elementen enthält, so kann dem aktuellen Rechenauftrag keine *jobID* eindeutig zugeordnet werden. Der Zugriff wird verweigert. Ist nur ein *ExecutionPolicy*-Element enthalten, wird der nächste Schritt ausgeführt.
3. Ist kein *ExecutionPolicy*-Element im Proxy-Credential enthalten, die *mandatory* Option jedoch gesetzt, ist der Zugriff zu verweigern. Andernfalls wird der nächste Schritt ausgeführt.
4. Die *jobID* liegt dem PDP nun vor. Über die gLite-API greift er auf den LB-Service zu und erfragt die dem Rechenauftrag zugeordneten Statusmeldungen. Ist dem LB-Service kein Rechenauftrag mit der referenzierten *jobID* bekannt, ist der Zugriff zu verweigern. Andernfalls wird der nächste Schritt ausgeführt.
5. Nun werden die Statusmeldungen durchlaufen. Befindet sich der Rechenauftrag – wie im Normalfall anzunehmen – im Zustand „Ready“, kann der nächste Schritt ausgeführt werden. Andernfalls ist der Zugriff zu verweigern.

6. Befand sich der Rechenauftrag zu einem früheren Zeitpunkt nicht in einem weitergehenden Zustand als „Ready“, wie z. B. „Running“, so wird der Zugriff gestattet. Wenn doch, muss es sich um einen „Resubmission“-Event handeln (vgl. Kapitel 5.5.1), der näher untersucht werden muss. Somit wird der nächste Schritt ausgeführt.
7. Ist der Rechenauftrag bereits einmal erfolgreich abgeschlossen worden (Zustand „Done (Success)“), so muss es sich um eine durch einen Angreifer wiederverwendete *jobID* handeln. Der Zugriff wird verweigert. Andernfalls wird der nächste Schritt ausgeführt.
8. Liegt ein „Resubmission“-Event vor? Wenn ja, wird der Zugriff gestattet, andernfalls verweigert.

5.5.4 Ablauf der modifizierten Zugriffskontrolle für Rechenaufträge

Der Ablauf der modifizierten Zugriffskontrolle am *Computing Element* ist in Abbildung 5.9 auf der nächsten Seite exemplarisch dargestellt. Dabei wird vorausgesetzt, dass der Zugriff rechtmäßig erfolgt, d. h. dass der Nutzer den Rechenauftrag am LB-Service registriert hat und das Proxy-Credential, das der Nutzer an das WMS delegiert hat, nicht kompromittiert wurde. In einem Proxy-Zertifikat innerhalb des Proxy-Credentials ist eine nutzerdefinierte Richtlinie vorhanden, die ein *Execution-Policy*-Element enthält. Weiterhin ist der Nutzer Mitglied einer Virtuellen Organisation, die ihm die Berechtigung zur Nutzung des CE erteilt hat und dazu auch berechtigt war. Der Nutzer ist lokal auf dem CE zum Zugriff berechtigt. Die Abgabe des Rechenauftrags läuft unter diesen Voraussetzungen wie folgt ab:

1. Der Nutzer registriert die *jobID* am LB-Service. Eine nachträgliche Änderung der *jobID* oder ihre Löschung ist nicht möglich.
2. Der Nutzer übergibt den Rechenauftrag an das WMS, das daraufhin ein passendes CE ermittelt.
3. Das WMS kontaktiert den *Gatekeeper* des CE zur gegenseitigen Authentifizierung und Autorisierung des Rechenauftrags.
4. Der *Gatekeeper* fordert eine Entscheidung über den Zugriffswunsch beim LCAS an.
5. Aus Effizienzgründen wird der PDP zuerst befragt, der anhand der lokalen Richtlinien entscheidet. Diese bestehen aus *Blacklists*, d. h. Listen mit Nutzern, denen der Zugriff verweigert wird. Eine derartige Entscheidung erfordert keine kryptographischen Operationen und verursacht daher wenig Rechenaufwand.
6. Nachfolgend wird die Entscheidung anhand der VO-Richtlinien durchgeführt. Hierfür werden die Attributzertifikate aus dem Proxy-Credential des Nutzers extrahiert und auf ihre Gültigkeit überprüft. Hat der Nutzer den Nachweis

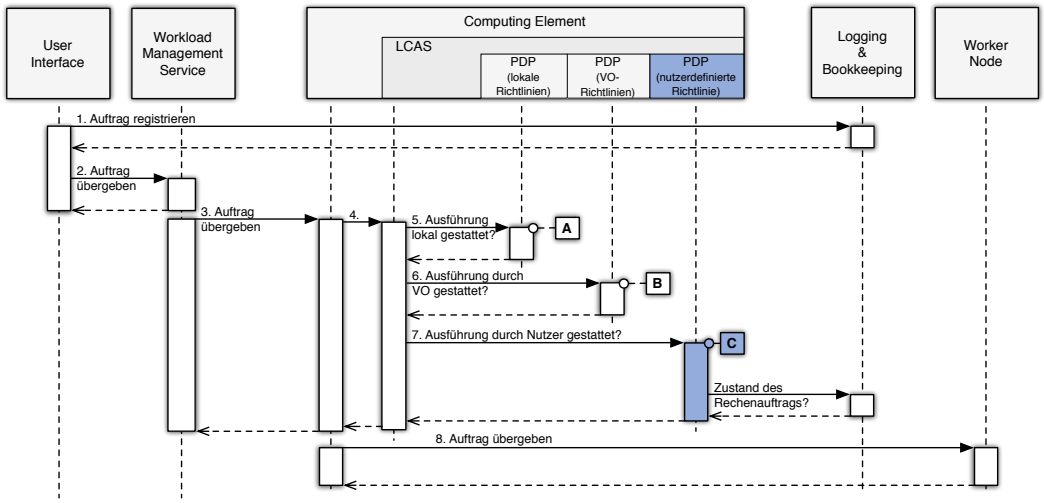


Abbildung 5.9: Abgabe eines Rechenauftrags (modifizierte Zugriffskontrolle)

über die Mitgliedschaft in einer unterstützten VO gebracht, wird der Zugriff gestattet.

7. Die letzte Zugriffsentscheidung erfolgt anhand der nutzerdefinierten Richtlinien. Dieser Vorgang ist bedingt durch die Abfrage des LB-Service und der notwendigen gegenseitigen Authentifizierung mit einem erhöhten Rechenaufwand verbunden. Daher erfolgt sie zum Schluss. Der genaue Ablauf ist Kapitel 5.5.3 zu entnehmen.
8. Nach Übermittlung der positiven Entscheidung durch das LCAS gewährt der *Gatekeeper* den Zugriff und leitet den Rechenauftrag an das LRMS weiter, das diesen nachfolgend an einen WN zur Bearbeitung übergibt.

5.6 Modifikation des MyProxy-Dienstes

Der Mechanismus der Erneuerung von Proxy-Zertifikaten zur Unterstützung von langlebigen Rechenaufträgen wurde in Kapitel 3.1.3.2 vorgestellt. Die Aufgabe des MyProxy-Dienstes in Grid-Infrastrukturen auf Basis der gLite-Middleware ist es, Rechenaufträgen neue Proxy-Zertifikate auszustellen, wenn die für die Beauftragung verwendeten ihre Gültigkeit verlieren. Dafür wird bei Eingehen eines Zugriffs durch ein autorisiertes WMS ein neues Proxy-Zertifikat von dem dem MyProxy-Dienst zur Verfügung stehenden abgeleitet und dem WMS zugänglich gemacht. Nach der Erneuerung steht dem entsprechenden Rechenauftrag ein unbeschränktes Proxy-Credential zur Verfügung, da das zur Ableitung verwendete Proxy-Credential des MyProxy-Dienst ebenfalls unbeschränkt ist. Die zur Verfügung stehenden Privilegien werden somit in dem Fall erheblich erweitert, in dem der Rechenauftrag vor der Erneuerung lediglich über ein beschränktes Proxy-Credential verfügte.

5.6.1 Vorüberlegungen

Eine Erweiterung der Rechenaufträgen zur Verfügung stehenden Privilegien durch Erneuerung ihrer Proxy-Zertifikate konterkariert das angestrebte Ziel dieser Arbeit. Angestrebt wird eine durchgehende Beschränkung der delegierten Privilegien auf den für die Durchführung der vorgesehenen Aufgaben notwendigen Mindestumfang. Dafür muss gewährleistet werden können, dass kein unbeschränktes Proxy-Credential die Heimateinrichtung des Nutzers verlässt, da hier in der Regel die notwendigen Einflussmöglichkeiten enden. Um dieses Ziel aufrechterhalten zu können, musste eine Modifikation der Verfahren zur Erneuerung von Proxy-Credentials vorgenommen werden. Dabei wird davon ausgegangen, dass die Heimateinrichtungen der Nutzer lediglich UI-Systeme und MyProxy-Dienste für ihre Nutzer betreiben. Unbeschränkte Proxy-Credentials dürfen folglich nur auf diesen beiden Komponenten vorliegen.

5.6.2 Implementierung und Darstellung der veränderten Abläufe

Die vorgenommenen Modifikationen im Ablauf der Erneuerung von Proxy-Zertifikaten beschränken sich auf interne Abläufe im MyProxy-Dienst. Sie sind in Abbildung 5.10 auf der nächsten Seite dargestellt. Die Hinterlegung eines Proxy-Credentials auf dem MyProxy-Dienst (Schritt 1) und die Abgabe von Rechenaufträgen (Schritt 2) wurden nicht verändert. Auch der auf dem WMS für die Erneuerung von Proxy-Zertifikaten zuständige *proxy-renewd* musste nicht modifiziert werden. Dies deckt sich mit dem in Kapitel 4.1 formulierten Ziel der Wahrung der Kompatibilität mit den bestehenden Komponenten und Protokollen.

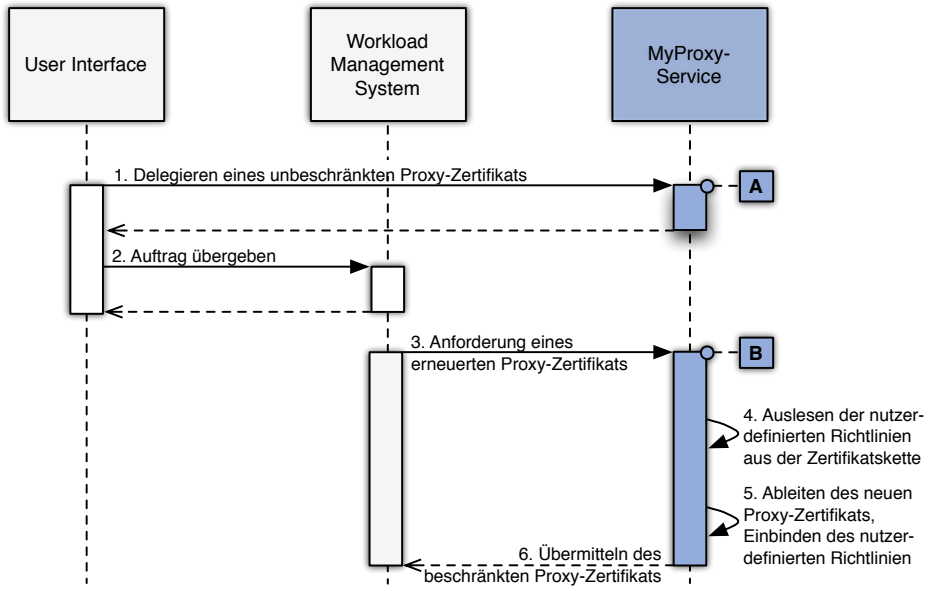


Abbildung 5.10: Proxy-Renewal durch den modifizierten MyProxy-Dienst

Wie bereits dargelegt wurde, läuft die Authentifizierung am MyProxy-Dienst bei Zugriffen für die Erneuerung von Proxy-Credentials zweistufig unter Verwendung sowohl des Host-Zertifikats des WMS als auch des zu erneuernden Proxy-Zertifikats des Nutzers ab. Damit liegen am MyProxy-Dienst die zur Beschränkung des Proxy-Zertifikats verwendeten nutzerdefinierten Richtlinien nach der erfolgreichen Authentifizierung und Autorisierung vor (Schritt 3).

Der MyProxy-Dienst wurde derart modifiziert, dass bei der Authentifizierung die vorgelegte Zertifikatskette durchlaufen wird und aus den enthaltenen Proxy-Zertifikaten die nutzerdefinierten Richtlinien extrahiert und zwischengespeichert werden (Schritt 4). Weiterhin wurde die Ableitung des neuen Proxy-Zertifikats vom unbeschränkten Proxy-Zertifikat des MyProxy-Dienstes um die Funktionalität ergänzt, die extrahierten nutzerdefinierten Richtlinien in das neu erstellte Proxy-Zertifikat einzubinden (Schritt 5). Schritt 6 der Abbildung entspricht wieder dem ursprünglichen Ablauf. In diesem wird die Delegation abgeschlossen, indem das Proxy-Zertifikat an das WMS weitergeleitet wird. Die neuen Funktionalitäten sind optional gestaltet, d. h. ihre Aktivierung kann über die zentrale Konfigurationdatei des MyProxy-Dienstes gesteuert werden.

Das Ziel der Modifikation des MyProxy-Dienstes, dass bei einem Zugriff zur Erneuerung eines beschränkten Proxy-Zertifikats auch nur ein entsprechend beschränktes Proxy-Zertifikat an das WMS ausgestellt wird, konnte erreicht werden. Eine Erweiterung der delegierten Privilegien ist damit ausgeschlossen.

6 Bewertung der Implementierung

6.1 Funktionsumfang

Das Ziel bei der Erweiterung der gLite-Middleware um Funktionalitäten der beschränkten Delegation war es, einen funktionsfähigen Prototypen bereitzustellen. Dabei sollten möglichst alle Funktionalitäten der betreffenden Komponenten der gLite-Middleware unterstützt bzw. nicht beeinträchtigt werden. Dieses Ziel konnte bei der Erweiterung der Komponenten UI, IO-Service sowie MyProxy-Service vollständig erreicht werden.

Das Web-Portal ist eine neue Komponente. Es bietet die Funktionalität, Jobs vom Typ „Normal“ unter Delegation beschränkter Proxy-Credentials zu beauftragen. Andere Arten von Rechenaufträgen sind aktuell nicht spezifizierbar. Da Grid-Infrastrukturen verstärkt für die Bearbeitung paralleler Rechenaufträge genutzt werden, sollte diese Funktionalität in zukünftigen Versionen implementiert werden.

Die erweiterten Zugriffskontrollmechanismen des CEs unterstützen Rechenaufträge vom Typ „Normal“ oder „MPI“. Diese werden durch den Nutzer am LB-Service unter einer *jobID* registriert, die in der Folge Bestandteil der zur Beschränkung der Delegation verwendeten *ExecutionPolicy* wird. *Job-Collections* oder Aufträge vom Typ „DAG“ verfügen jedoch über mehrere *JobIDs*, eine für jeden Sub-Job sowie eine für den gesamten Auftrag. Für eine beschränkte Delegation müssen folglich mehrere *ExecutionPolicy*-Elemente in dem zur Delegation verwendeten Proxy-Zertifikat enthalten sein. In Version 3.0 der gLite-Middleware verfügt das LCAS jedoch über keine Informationen über die *jobID* des aktuellen Rechenauftrags. Daher ist es dem PDP nicht möglich, festzustellen, über welchen der in den *ExecutionPolicies* referenzierten Rechenaufträgen aktuell zu entscheiden ist. Die Implementierung der Beschränkung von Delegation für *Job-Collections* und *DAGs* erfordert eine Änderung der gLite-Middleware. Eine Abhilfe ist jedoch durch eine Aufteilung dieser Rechenaufträge in mehrere einzelne vom Typ „Normal“ bzw. „MPI“ gegeben.

6.2 Sicherheitsaspekte

Der Ansatz sowie die darauf aufbauende Implementierung der beschränkten Delegation zielen darauf ab, die unautorisierte Nutzung kompromittierter Proxy-Credentials einzuschränken und sie im besten Fall komplett zu unterbinden. Es sind jedoch Szenerien denkbar, wenn auch extrem unwahrscheinlich, in denen ein Missbrauch dennoch möglich ist.

6.2.1 Angriffsvektor zur unautorisierten Ausführung von Jobs

Wie in Kapitel 3.2.3 dargelegt, ist die Kompromittierung von Grid-Ressourcen durch gezielte Angriffe möglich, wenn auch bei entsprechender Konfiguration unwahrscheinlich. Eine derartige Kompromittierung kann genutzt werden, um ein Proxy-Credential trotz Beschränkung durch eine eingebettete *ExecutionPolicy* zur Ausführung eines Rechenauftrags zu missbrauchen. Dafür sind folgende beiden Bedingungen zwingend zu erfüllen:

1. Der Rechenauftrag des Angreifers muss vor Ankunft des autorisierten Rechenauftrags des Nutzers auf dem CE an ein CE abgegeben werden, damit sein Auftrag als valide anerkannt wird. Die Zeitspanne, die dem Angreifer für einen derartigen Angriff zur Verfügung steht, variiert je nach Auslastung des WMS und der für die Beauftragung des Jobs zu übertragenden Datenmenge. Sie beläuft sich im Regelbetrieb auf deutlich unter fünf Minuten, typisch sind eine bis zwei Minuten.
2. Der Angreifer muss Zugriff auf das beschränkte Proxy-Credential des Nutzers haben. Dieses Proxy-Credential befindet sich vor Eintreffen des Auftrags auf dem CE neben dem UI des Nutzers lediglich noch auf dem WMS, das den Auftrag an ein passendes CE weiterleitet. Da Grid-Nutzer über keinen lokalen Zugriff auf das WMS verfügen, muss ein entsprechender Angriff zur Kompromittierung delegierter Proxy-Credentials durch einen erfolgreichen Angriff auf einen Netzdienst erfolgen, um Zugriff auf lokale Dateien zu erhalten. Weiterhin benötigt der Angreifer lokale Administratorprivilegien auf dem WMS.

Ein unter den beiden genannten Bedingungen erfolgreicher Angriff kann als sehr unwahrscheinlich, jedoch nicht als unmöglich eingestuft werden. Der Nutzer kann einen derartigen Missbrauch durch Abfrage des aktuellen Zustands am LB-Server erkennen, da bereits nach kurzer Zeit Versuche zur erneuten Beauftragung des Jobs an ein anderes CE durch das WMS gestartet werden. In einem solchen Fall kann der Nutzer durch Abbrechen des Rechenauftrags den unrechtmäßig beauftragten Job beenden und erneut einen Rechenauftrag starten.

6.2.2 Angriffsvektor zum unautorisierten Zugriff auf Speicherobjekte

Im Gegensatz zum verbleibenden Angriffsvektor für die unautorisierte Ausführung von Rechenaufträgen ist die Beschränkung durch in Proxy-Zertifikate eingebettete *FilePolicies* ohne Einschränkung wirksam. Es gibt keinen Zeitraum nach ihrer Erstellung, in dem eine Erweiterung der Beschränkung möglich ist. Es kann jedoch durch Kompromittierung entsprechender Proxy-Credentials zu Zugriffen auf Speicherobjekte kommen, die durch die *FilePolicies* gestattet sind. Durch einen derartigen Angriff ist es z. B. möglich, dass ein Angreifer Dateien von einem SE abruft.

6.3 Performance der implementierten Infrastrukturkomponenten

Die Einführung zusätzlicher Komponenten in bestehende Sicherheitsarchitekturen und die damit einhergehende Erhöhung der Komplexität verursacht einen Mehraufwand in der Bearbeitung von Anfragen. Für die ordnungsgemäße Funktion des Gesamtsystems darf dieser im Normalbetrieb keine Einschränkung verursachen. Auch sollte er nicht für erfolgreiche Dienstverweigerungsangriffe ausgenutzt werden können, die mittels zahlreicher, gleichzeitiger Anfragen an das System durchgeführt werden.

Die Performance der implementierten Komponenten ist abhängig von ihrer Position in der Grid-Infrastruktur unterschiedlich bedeutsam. Bei Komponenten, die durch die Heimateinrichtungen der Nutzer administriert werden und bei Bedarf durch gleichgeartete Systeme ergänzt werden können, ist sie von untergeordneter Bedeutung. Dies betrifft sowohl das erweiterte *User Interface* als auch den MyProxy-Dienst. Beide können bei steigenden Nutzerzahlen einer Einrichtung und damit einhergehender Erhöhung der Arbeitslast der entsprechenden Systeme mit geringem Aufwand durch gleichartige Systeme unterstützt werden. Ein solcher Ansatz ist bei Infrastrukturkomponenten, die von Anwendergruppen mehrerer Einrichtungen genutzt werden, nur mit erheblichem Administrationsaufwand möglich, der zudem mit einer entsprechenden Erhöhung der Komplexität der gesamten Infrastruktur einhergeht.

Um den durch die Erweiterung entstandenen Mehraufwand bewerten zu können, wurden die Laufzeiten der Zugriffskontrollmechanismen des IO-Service sowie des CE ermittelt. Diese wurden zum Einen im lokalen Kontext bewertet, d. h. durch Vergleich bestehender mit neu hinzugekommenen Autorisierungskomponenten. Zum Anderen wurde eine Bewertung des Mehraufwands im Kontext der gesamten Infrastruktur vorgenommen, um ein aussagekräftiges Gesamtbild zu erhalten.

6.3.1 Laufzeiten der Zugriffskontrolle im IO-Service

Die Messungen zur Ermittlung der Laufzeiten der Zugriffskontrollmechanismen im erweiterten IO-Service wurden unter Verwendung der in Anhang A.1 vorgestellten Systeme *ioadmin1* sowie *ui1* durchgeführt. Zusätzlich wurde ein *FireMan Data Catalog* verwendet, der in einer virtuellen Maschine (VM) auf Basis der Virtualisierungsumgebung Xen installiert war. Die zugrunde liegende Hardware bestand aus einem Server mit zwei Intel Xeon (Clovertown) Quad-Core Prozessoren und 16 GB Arbeitspeicher. Der VM waren zwei virtuelle CPU-Kerne zugeteilt sowie 1 GB Arbeitsspeicher.

Zur Ermittlung der Laufzeiten wurden je Messung vier Zeitnahmen durchgeführt. Die erste Zeitnahme erfolgte bei Aufruf der Zugriffskontrolle (t_1, vgl. Schritte 2. und 3. in Abbildung 5.5 auf Seite 102), die zweite Zeitnahme bei Aufruf des PDP

auf Basis der nutzerdefinierten Richtlinien (t_2), die dritte bei dessen Beendigung (t_3), die vierte am Ende der gesamten Zugriffskontrolle (t_4). Aus diesen Zeitnahmen wurden durch Differenzbildung zwei Laufzeiten ermittelt:

$$t_{ubp-io} = t_3 - t_2 \tag{6.1}$$
$$t_{authz-io} = t_4 - t_1 \tag{6.2}$$

Laufzeit t_{ubp-io} ist dabei der Zeitraum, den das System für die Abarbeitung des hinzugefügten PDP benötigt, $t_{authz-io}$ umfasst die gesamte zur Zugriffskontrolle eines Verbindungswunsches benötigte Zeit.

Die Zeitnahmen erfolgten durch Auslesen des *Time Stamp Counters* (TSC) des Hauptprozessors des Systems. Dieser wird bei Systemstart initialisiert und mit jedem Prozessortakt inkrementiert. Damit verfügt der TSC über die höchste zeitliche Auflösung aller Uhren in einem herkömmlichen UNIX-System auf Basis von PC-Hardware. Die Ausgaben des TSC geben die Anzahl der Taktzyklen an, die seit Systemstart vergangen sind und erfolgen als Ganzzahlwert mit 64 Bit Länge. Ein Überlauf der TSC konnte ausgeschlossen werden und wurde im Programm nicht berücksichtigt. Die Umrechnung der Ergebnisse in Millisekunden erfolgte nachfolgend durch Division der Zeitangaben in Taktzyklen durch den Prozessortakt, der bei Systemstart durch den Betriebssystemkern ermittelt wurde:

$$t_{ubp-io}[ms] = 1000 \cdot \frac{t_{ubp-io}[\text{TSC Zyklen}]}{2,7996 \cdot 10^9[\frac{TSC Zyklen}{s}]} \tag{6.3}$$

Eine Quelle für Ungenauigkeiten liegt bei derartigen Messungen in unerwünschten Taskwechseln. Diese werden durch den Prozess-Scheduler des Betriebssystems verursacht und sorgen für eine Unterbrechung des Programmablaufs. Da der TSC während der Unterbrechung des IO-Service fortlaufend inkrementiert wird, werden die Ergebnisse entsprechend verfälscht. Messungen, während denen ein Taskwechsel erfolgte, lassen sich jedoch leicht anhand der erheblichen Abweichungen der Ergebnisse von gleichartigen Messungen erkennen und als Ausreißer aus der Auswertung ausschließen.

Die Messungen erfolgten unter Variation der Anzahl n_{fp} elementarer *FilePolicy*-Elemente in der nutzerdefinierten Richtlinie, die in dem zur Authentifizierung verwendeten Proxy-Zertifikat enthalten war. Es wurden Messungen durchgeführt mit einer, zwei, fünf sowie 10, 15 und 20 elementaren *FilePolicy*-Elementen. Die Verwendung von mehr als 20 *FilePolicy*-Elementen in einer Richtlinie ist nicht mehr als sinnvoll anzusehen. Hier sollten Nutzer von der Option zur Referenzierung von Zielobjekten mittels *Wildcards* Gebrauch machen (vgl. Kapitel 5.2.1.1).

Um einen aussagekräftigen Mittelwert zu erhalten, wurden jeweils 100 Abläufe mit derselben Anzahl n_{fp} von *FilePolicy*-Elementen vermessen. Ergebnisse, die um

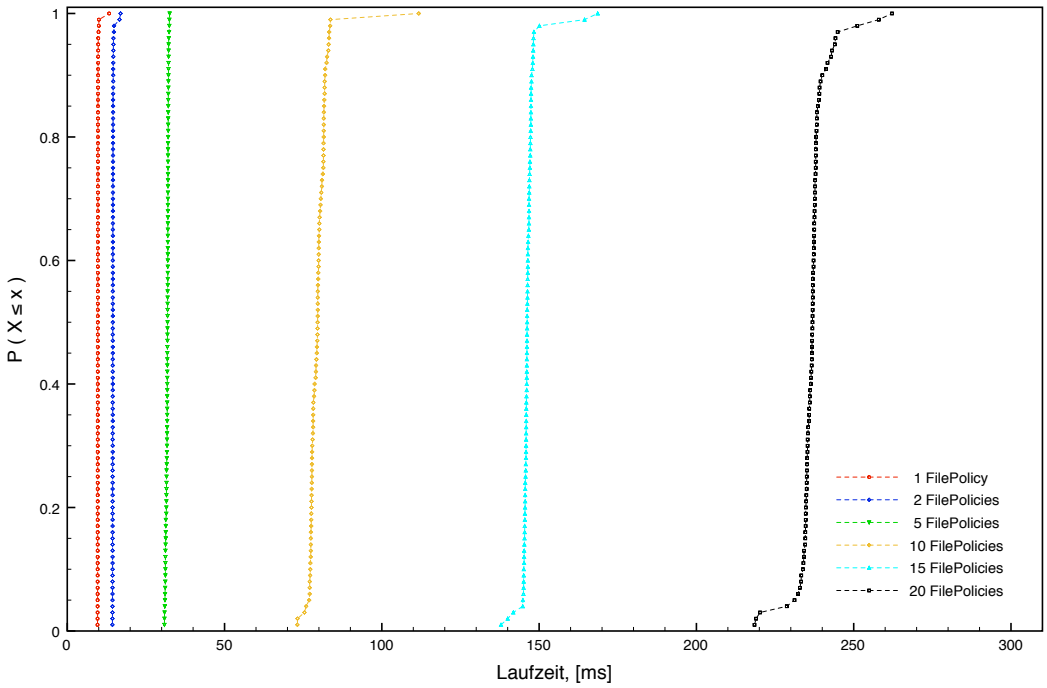


Abbildung 6.1: Laufzeiten $t_{ubp-io}(n_{fp})$ des hinzugefügten PDP im IO-Service

mehr als das Fünffache vom Median der Messreihe abwichen, wurden dabei als Ausreißer gewertet, die durch unerwünschte Taskwechsel verursacht wurden. Diese Ausreißer wurden aus der Mittelwertbildung ausgeschlossen, da sie andernfalls zu erheblichen Verfälschungen geführt hätten.

In Abbildung 6.1 sind die Laufzeiten t_{ubp-io} des hinzugefügten PDP in Form von kumulierten Verteilungsfunktionen (CDF) in Abhängigkeit von n_{fp} aufgetragen. Aus den Verteilungsfunktionen lässt sich eine hohe Stabilität der Messungen ablesen, die eine Mittelwertbildung rechtfertigt. Es mussten keine Ergebnisse als Ausreißer klassifiziert und entfernt werden. Die Mittelwerte der Laufzeiten des PDP sind in Tabelle 6.1 aufgeführt. Aus diesen ist ein nicht-linearer Anstieg der Laufzeiten bei Erhöhung von n_{fp} zu entnehmen.

$$\Delta t_{ubp,fp}(n_{fp}-1) = \frac{t_{ubp-io}(n_{fp}) - t_{ubp-io}(1)}{n_{fp}-1} \tag{6.4}$$

Ermittelt man die Veränderung des Anstiegs der Laufzeit je hinzugefügtem *FilePolicy*-Element anhand Gleichung 6.4 für die einzelnen Messreihen (vgl. Abbildung 6.2 auf der nächsten Seite) ergibt sich ein annähernd linearer Zusammenhang zwischen

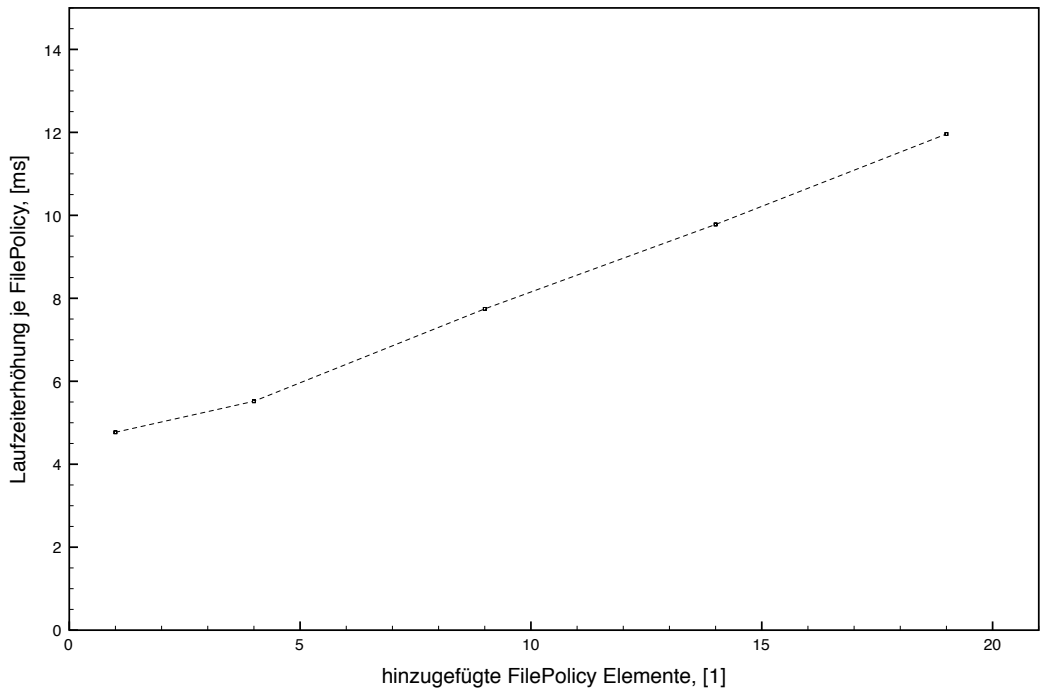


Abbildung 6.2: Laufzeitzuwachs des hinzugefügten PDP je hinzugefügtem FilePolicy-Element

der Anzahl der über das erste *FilePolicy*-Element hinaus hinzugefügten *FilePolicy*-Elemente und dem Zuwachs von $t_{ubp-io}(n_{fp} - 1)$ je hinzugefügtem *FilePolicy*-Element. Es lässt sich eine Steigung von $\approx 0,4\frac{ms}{n_{fp}-1}$ beobachten.

Dieses Verhalten ist durch die Implementierung des PDP zu erklären. Innerhalb des PDP wird nach dem Aufruf in einem ersten Durchgang der nutzerdefinierten Richtlinie ein *myPolicies*-Objekt aufgebaut, das in einem zweiten Durchgang nach Regeln durchsucht wird, die auf den gewünschten Zugriff zutreffen. Somit wächst der Aufwand für die Abarbeitung der gesamten nutzerdefinierten Richtlinie nichtlinear mit n_{fp}.

Die durch die Bearbeitung der Richtlinien im hinzugefügten PDP entstehenden Verzögerungen sind im Kontext der Laufzeit der gesamten Zugriffskontrolle des IO-Service zu bewerten. Die Auswertung der entsprechenden Messreihen wurde nach dem gleichen Verfahren wie für die Laufzeiten des PDP durchgeführt. Abweichend davon wurden jedoch in jeder Messreihe Ausreißer entfernt, um aussagekräftige Mittelwerte zu erhalten (vgl. Abbildung 6.3 auf der nächsten Seite und Tabelle 6.1). Eine Begründung für das erhöhte Auftreten von Ausreißern ist in zwei Ursachen zu suchen. Zum Einen ist die Wahrscheinlichkeit von Taskwechseln durch die deutlich

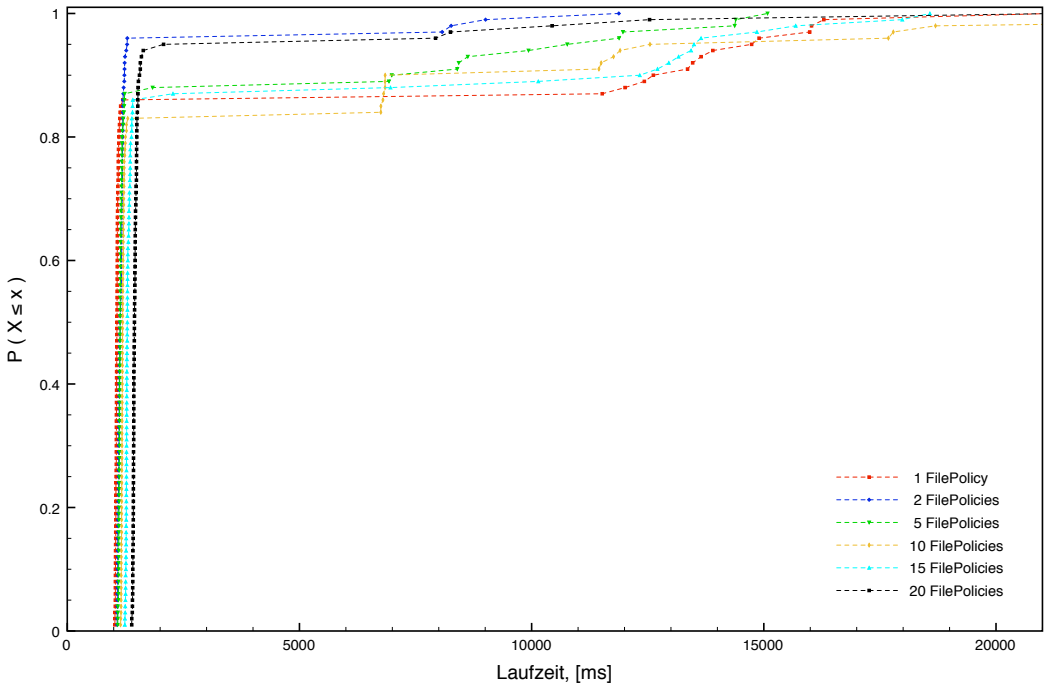


Abbildung 6.3: Laufzeiten $t_{authz-io}(n_{fp})$ der Zugriffskontrolle im IO-Service

längere Laufzeit der übrigen Zugriffskontrolle des IO-Service größer. Zum Anderen ist die Antwortverzögerung des FireMan-Service, der in diesem Abschnitt der Zugriffskontrolle kontaktiert wird, nicht statisch. Da durch beide Ursachen eine im Verhältnis zur Gesamtlaufzeit der Zugriffskontrolle bessere Performance des PDP suggeriert wird, wurden die Ausreißer entfernt.

Nach Entfernen der Ausreißer stellt sich das Verhältnis der Laufzeit des PDP auf Basis der nutzerdefinierten Richtlinien zur Gesamtlaufzeit der Zugriffskontrolle im IO-Service mit einem Anteil von 16,19 % bei der Einbettung von 20 *FilePolicy*-Elementen in die nutzerdefinierte Richtlinie als gering dar. Diese Feststellung gilt verstärkt für die Einbettung einer niedrigeren Anzahl von *FilePolicy*-Elementen, für die der Anteil der Laufzeit des hinzugefügten PDP entsprechend sinkt. Auch sollte die Verwendung einer geringen Anzahl von *FilePolicy*-Elementen in den nutzerdefinierten Richtlinien die Regel sein, da sich für die Adressierung einer großen Anzahl von Zielobjekten die Verwendung von *Wildcards* empfiehlt.

Auch absolut betrachtet nimmt sich der Performanceverlust gering aus, da Speicherressourcen in Grid-Infrastrukturen für die Ablage großer Datenmengen im Bereich von Gigabyte in einzelnen Dateien optimiert sind. So stellt sich der aus der

Anzahl n_{fp} FilePolicy Elemente	mittlere Laufzeit PDP $t_{ubp-io}(n_{fp})$ [ms]	mittlere Laufzeit Zugriffskontrolle $t_{authz-io}(n_{fp})$ [ms]	Anteil PDP [Prozent]
1	9,72	1069,04	0,91
2	14,49	1147,31	1,26
5	31,79	1147,06	2,77
10	79,67	1192,95	6,68
15	146,61	1311,01	11,18
20	236,9	1463,03	16,19

Tabelle 6.1: Mittlere Laufzeiten der Zugriffskontrollmechanismen des IO-Service

mittleren Laufzeit t_{ubp-io} und einer idealisierten Übertragung von 1 GB[1] über eine Datenverbindung mit einer Transferrate von netto 1 GBit/s errechnete Overhead (vgl. Formel 6.5) mit weniger als 3 % gering dar (vgl. Tabelle 6.2). Dabei werden Verzögerungen durch die Anwendung weiterer Authentifizierungs- und Autorisierungsverfahren oder des *Slow Start* Algorithmus des TCP nicht berücksichtigt. Bei Berücksichtigung dieser Verzögerungen reduziert sich der prozentuale Performanceverlust entsprechend.

$$O[Prozent] = 100 \cdot \frac{\frac{D[byte] \cdot 8 \frac{bit}{byte}}{R[\frac{bit}{s}]} + t_{ubp-io}(n_{fp})[s]}{\frac{D[byte] \cdot 8 \frac{bit}{byte}}{R[\frac{bit}{s}]}} \tag{6.5}$$

Anzahl n_{fp} FilePolicy Elemente	Overhead O [Prozent] bei Übertragung mittels $R = 1Gbps$ von D 10 GB	5 GB	1 GB
1	0,01	0,02	0,12
2	0,02	0,04	0,18
5	0,04	0,08	0,40
10	0,10	0,20	1,00
15	0,18	0,37	1,83
20	0,30	0,59	2,96

Tabelle 6.2: Overhead durch erweiterte Zugriffskontrollmechanismen des IO-Service

[1] Zur Vereinfachung wird 1Gb als 10^9 Bit angenommen und nicht als 2^{20} Bit.

6.3.2 Laufzeiten der Zugriffskontrolle im Computing Element

Die Messungen zur Ermittlung der Laufzeiten der Zugriffskontrolle im CE wurden auf der in Anhang A.1 vorgestellten Testumgebung durchgeführt. Dabei fanden ein gLite-CE auf *ce2*, ein WMS auf *wms1*, ein UI auf *ui1* sowie der LB-Service auf *lb1* Verwendung.

Äquivalent zu den Messungen auf dem IO-Service wurden je Messung vier Zeitnahmen zur Ermittlung der Laufzeiten der Zugriffskontrollmechanismen durchgeführt. Der erste Messpunkt (t_1) befand sich direkt nach dem Aufruf des LCAS durch den *Gatekeeper* (vgl. Schritte 5-7 in Abbildung 5.9 auf Seite 110). Die zweite (t_2) und dritte Zeitnahme (t_3) erfolgte nach Aufruf bzw. vor Beendigung des PDP auf Basis der nutzerdefinierten Richtlinien, die Zeitnahme t_4 erfolgte unmittelbar vor Beendigung des LCAS. Aus diesen Zeitnahmen wurden zwei Laufzeiten ermittelt:

$$t_{ubp-ce} = t_3 - t_2 \tag{6.6}$$

$$t_{lcas} = t_4 - t_1 \tag{6.7}$$

Auch hier entspricht die erste Laufzeit t_{ubp-ce} dem Zeitraum, den das System für die Abarbeitung des hinzugefügten PDP benötigt, t_{lcas} umfasst die gesamte zur Zugriffskontrolle eines Verbindungswunsches benötigte Zeit. Die Zeitnahmen erfolgten durch Auslesen des TSC, die Umrechnung in Sekunden wurde durch Division der ausgelesenen Werte durch die Taktfrequenz des Hauptprozessors von *ce2* durchgeführt.

Da die Messungen unter möglichst identischen Bedingungen durchgeführt werden sollten, wurden synthetische Rechenaufträge minimalen Umfangs in der unbelasteten Testumgebung beauftragt. Im Gegensatz zu den in Kapitel 6.3.1 vorgestellten Performancemessungen am IO-Service entfällt die Variation der Anzahl elementarer Richtlinien n_{fp} in der eingebetteten *Execution Policy*, da ein Proxy-Zertifikat nur jeweils eine elementare Richtlinie für die Beschränkung der Privilegien zur Ausführung eines Rechenauftrags enthalten darf.

Zur Ermittlung eines aussagekräftigen Mittelwerts für t_{ubp-ce} und t_{lcas} wurden 100 Messungen durchgeführt, die einzelnen Messungen wurden jeweils nach Abschluss der vorhergehenden gestartet, um die Belastung des Systems nicht zu erhöhen und damit die Ergebnisse zu verfälschen. Die ermittelten Laufzeiten der Einzelmessungen sind in Abbildung 6.4 auf der nächsten Seite sowohl für den PDP auf Basis der nutzerdefinierten Richtlinien (t_{ubp-ce}) als auch für den LCAS gesamt (t_{lcas}) in Form von kumulierten Verteilungsfunktionen dargestellt.

Wie der Abbildung zu entnehmen ist, traten im Gegensatz zu den Laufzeitmessungen am IO-Service keine durch Taskwechsel verursachte Ausreißer auf, daher konnten alle Messergebnisse für die Mittelwertbildung der Laufzeiten verwendet

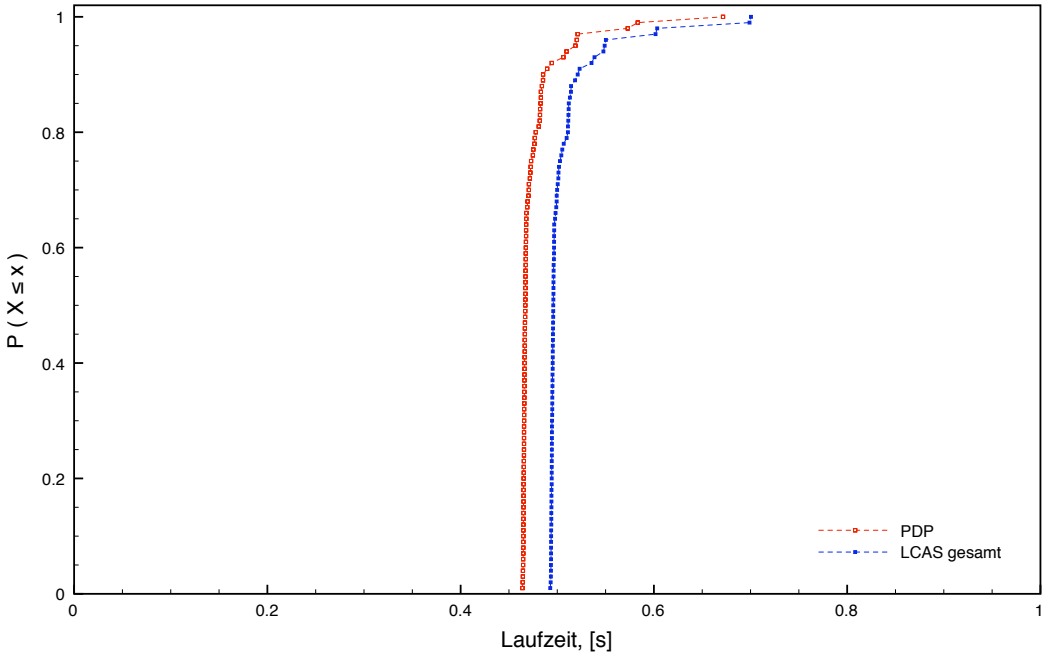


Abbildung 6.4: Laufzeiten der Zugriffskontrollmechanismen im CE

werden (vgl. Tabelle 6.3). Aus den Ergebnissen lässt sich eine deutliche Steigerung der Laufzeit der gesamten Zugriffskontrolle ableiten. Diese resultiert unmittelbar aus der Implementierung des hinzugefügten PDP, der für die Entscheidung über die Gewährung oder Ablehnung des Zugriffs jeweils eine gegenseitig authentifizierte Abfrage des LB-Service durchführen muss.

Im typischen Produktivbetrieb äußert sich die zusätzliche Belastung der beteiligten Systeme auf zwei Arten. Zum Einen erhöht sich die Latenz von der Beauftragung eines Jobs bis zu dessen Übergabe an das LRMS. Eine begleitende Messung dieser Latenz durch Auswertung der Statusmeldungen des UI und des CE direkt am LB-Service mit dem Werkzeug *tcpdump* ergab eine mittlere Latenz von $51,2s$, so dass der Anteil der Zugriffskontrolle des CE an der Latenz unter 1% betrug und somit vernachlässigt werden kann. Zum Anderen wird die Arbeitslast sowohl auf dem CE als auch auf dem LB-Service erhöht. Die Arbeitslast auf dem CE durch den hinzugefügten PDP besteht im Wesentlichen aus der Abfrage des LB-Service und den damit verbundenen Authentifizierungsvorgängen. In einer unmodifizierten gLite-Infrastruktur erfolgen drei Statusmeldungen je Rechenauftrag durch das CE an den LB-Service, so dass die diesbezügliche Belastung durch den PDP um 33% erhöht wird. Im Gesamtsystem CE stellt die Kommunikation mit dem LB-Service jedoch nur einen geringen Anteil an der gesamten Arbeitslast (vgl. Kapitel 2.1.3.1)

dar, so dass die anteilige Belastung durch den PDP erheblich geringer ist. Weiterhin muss berücksichtigt werden, dass Rechenaufträge in Grid-Infrastrukturen üblicherweise Bearbeitungsdauern von mehreren Stunden haben, so dass selbst ein CE, an das eine große Zahl von *Worker Nodes* angeschlossen ist, nur mit einer geringen Anzahl von Rechenaufträgen pro Minute zu rechnen hat. Für den LB-Service erhöht sich die Belastung von sechs Authentifizierungsvorgängen auf sieben ebenfalls je erfolgreich abgeschlossenem Rechenauftrag. Nach Aussagen der Entwickler von *gLite* ist der LB-Service in der verwendeten Version in der Lage, pro Tag ca. 27000 Rechenaufträge zu verwalten [Gra07], so dass selbst im schlechtesten Fall bei direkter Skalierung mit der Anzahl der Authentifizierungsvorgänge noch von mehr als 23000 Rechenaufträgen pro Tag und LB-Service ausgegangen werden kann.

Für Dienstverweigerungsangriffe auf das Zugriffskontrollsystem des CE muss der Angreifer über ein valides Zertifikat einer durch den CE akzeptierten CA verfügen, da der Angriff andernfalls bereits bei der Authentifizierung scheitert. Um eine signifikante Last durch den hinzugefügten PDP zu erzeugen, muss es sich weiterhin um ein Proxy-Zertifikat eines autorisierten Nutzers handeln, das zudem über eine eingebettete *ExecutionPolicy* verfügt. Somit muss der Angreifer bereits im Vorhinein eine andere Grid-Ressource kompromittiert haben, auf der derartige Proxy-Credentials vorliegen. Beide Voraussetzungen lassen es unwahrscheinlich erscheinen, dass sich ein Angreifer für einen Dienstverweigerungsangriff auf die Grid-Infrastruktur diese Komponente als Angriffsziel aussucht. Sollte sich dieses Szenario dennoch als Problem darstellen, kann der PDP derart verändert werden, dass in einem Cache Kombinationen aus jobIDs einschließlich DN der beantragenden Nutzer vorgehalten werden, die in einem kurzen Zeitraum für eine große Anzahl von Autorisierungsanfragen verwendet wurden. Für diese wäre die Autorisierung ohne Abfrage des LB-Service abzulehnen, so dass die Last am CE durch den Angriff nur unwesentlich erhöht würde.

Abschließend lässt sich feststellen, dass die Belastung durch den PDP auf Basis der nutzerdefinierten Richtlinien im Verhältnis zu der bisherigen Belastung durch das bestehende Zugriffskontrollsystem zwar deutlich steigt. Bei globaler Betrachtung ist diese jedoch weder für den Nutzer in Hinblick auf die Bearbeitungsdauer seiner Rechenaufträge noch für den Administrator in Hinblick auf den erreichbaren Durchsatz an Rechenaufträgen signifikant.

Laufzeit PDP t_{ubp-ce} [s]	Laufzeit Zugriffskontrolle t_{lcas} [s]	Anteil PDP [Prozent]
0,471	0,506	93,1

Tabelle 6.3: Mittlere Laufzeiten der Zugriffskontrollmechanismen im CE

7 Zusammenfassung und Ausblick

Der Beitrag dieser Arbeit liegt in der Konzeption eines Ansatzes zur effektiven Beschränkung delegierter Privilegien, durch dessen Implementierung eine grundlegende Schwäche im Sicherheitsmodell heutiger Grid-Infrastrukturen entscheidend reduziert werden kann. Die Konzentration des Ansatzes auf Privilegien zum Zugriff auf Zielobjekte der Grid-Ebene setzt das Grid-Paradigma konsequent um und dient damit der Entlastung des Anwenders von detaillierten Kenntnissen über die genutzten Ressourcen und den auf diesen vorliegenden Objekten. Darüber hinaus konnte im Rahmen dieser Arbeit gezeigt werden, dass die Adressierung von sowohl statischen als auch dynamischen Zielobjekten in Delegationen praktikabel ist und eine darauf aufbauende Zugriffskontrolle die Beschränkung der delegierten Privilegien effektiv durchsetzen kann. Durch die nutzerdefinierte Spezifikation der delegierten Privilegien wird der Aufwand für die Verwaltung von Attributautoritäten im erarbeiteten Ansatz konstant gehalten. Damit werden neue Quellen für Fehlkonfigurationen und daraus entstehende Sicherheitslücken wirksam vermieden. Weiterhin wird die Kompatibilität zu den bestehenden Autorisierungsmechanismen durch die Einbindung der spezifizierten Richtlinien in Proxy-Zertifikate gewahrt, wodurch eine erleichterte Integration der auf den Ansatz aufbauenden Implementierungen in bestehende Umgebungen erreicht wird. Der Verzicht auf zusätzliche zentrale Dienste zur Unterstützung der Delegationsfunktionalität vermeidet eine Steigerung der Komplexität der Grid-Infrastruktur und damit weitere potentielle Fehlerquellen.

Die Basis für die Erarbeitung des Ansatzes bildet eine Analyse der Autorisierungsmechanismen in Grid-Infrastrukturen. Diese werden exemplarisch anhand der gLite-Middleware des europäischen EGEE-Projekts durchgeführt, die auf dem *Globus Toolkit* basiert. Die bestehenden Mechanismen zur Delegation von Nutzerprivilegien werden in Hinblick auf Funktionalität und Sicherheit untersucht und Schwächen identifiziert. Nachfolgend werden allgemeingültige Kriterien für Ansätze zur beschränkten Delegation aufgestellt und begründet sowie bestehende Ansätze analysiert und auf Basis dieser Kriterien bewertet.

Aus der Analyse werden Anforderungen an einen umfassenden Ansatz zur beschränkten Delegation von Privilegien abgeleitet und formuliert. Die Methodik des Ansatzes wird hinsichtlich der agierenden Entitäten, der Dimension der Beschränkung sowie der unterstützten Objekte dargelegt und begründet. Aus der Methodik werden die benötigten Bestandteile der nutzerdefinierten Richtlinien abgeleitet. Für deren Transport zu den Grid-Ressourcen wird das Push-Modell, als Transportmedium Proxy-Zertifikate verwendet. Die für die Durchsetzung der Richtlinien benötigten Mechanismen der Zugriffskontrolle werden nachfolgend diskutiert.

Zudem werden Verfahren für eine effiziente Unterstützung der Nutzer bei der Erstellung der für die beschränkte Delegation benötigten Richtlinien diskutiert.

Die prototypische Implementierung des erarbeiteten Ansatzes erfolgt am Beispiel der gLite-Middleware, sie ist jedoch prinzipiell auf alle Grid-Middlewares portierbar, die Proxy-Zertifikate zur Delegation von Privilegien einsetzen. Die Implementierung unterstützt je einen Dienst der gLite-Middleware als Beispiel für statische bzw. dynamische Zielobjekte. Erstere werden durch Dateien und Verzeichnisse auf Speicherressourcen repräsentiert, letztere durch Rechenaufträge auf *Computing Element*-Systemen. Die Kodierung der nutzerdefinierten Richtlinien erfolgt in XACML, wodurch eine Erweiterbarkeit für zukünftig unterstützte Dienste einfach möglich ist. Der Nutzer wird bei der Erstellung von Richtlinien unter Anwendung der in Kapitel 4 diskutierten Methoden unterstützt. Die automatisierte Erstellung der benötigten Richtlinien kommt bei der Beschränkung der Ausführung von Rechenaufträgen zum Einsatz. Für die Unterstützung bei der Erstellung von Richtlinien zur Beschränkung des Zugriffs auf Speicherressourcen wurde eine grafische Nutzerschnittstelle implementiert, die eine intuitive Auswahl von Zielobjekten und Zugriffsarten auf diese ermöglicht.

Die Implementierung erfüllt die im Ansatz formulierten Anforderungen in Hinblick auf Funktionalität. Das erreichte Sicherheitsniveau stellt eine erhebliche Verbesserung gegenüber einer gLite-Umgebung ohne Funktionalität zur Beschränkung delegierter Privilegien dar. Ein theoretisch verbleibender Angriffsvektor auf die Beschränkung der Privilegien zur Ausführung von Rechenaufträgen setzt eine vorhergehende Kompromittierung einer Grid-Ressource voraus und ist zudem nur in einem engen Zeitfenster wirksam. Zur Bewertung der Performance wurden die implementierten Infrastrukturkomponenten in einer Testumgebung am RRZN eingehend untersucht. Die Ergebnisse der Messungen lassen das erreichte Niveau als für eine produktive Umgebung ausreichend erscheinen.

7.1 Ausblick

Die Förderung von Grid-Projekten wie dem europäischen EGEE-II oder dem deutschen D-Grid lassen den politischen Willen erkennen, leistungsfähige nationale sowie internationale eScience-Verbünde aufzubauen und langfristig zu betreiben. Diese Integration umfasst in zunehmenden Maße Anwendergruppen, die nicht den klassischen Nutzern von Großrechnern zuzuordnen sind. So ist ein mit der Förderung der D-Grid-Projekte verbundenes Ziel des deutschen Bundesministeriums für Bildung und Forschung, weitere Wissenschaftsbereiche aber auch kommerzielle Anwender an einer integrierten deutschen eScience-Infrastruktur zu beteiligen.

Aus diesen Entwicklungen lässt sich eine generell wachsende Bedeutung von Grid-Infrastrukturen ableiten, die jedoch mit einem erhöhten Sicherheitsbedürfnis verknüpft ist. Während die Anwender der ersten Grid-Verbünde im Wesentlichen

aus der Hochenergiephysik entstammten und keinerlei datenschutzrechtliche Vorgaben zu beachten hatten, sind die neuen Anwendergruppen aus dem biologisch-medizinischen oder ingenieurwissenschaftlichen Umfeld entweder durch rechtliche Vorgaben oder aus Eigeninteresse an einem Schutz ihrer Daten interessiert. Weiterhin wird, bedingt durch den Zugriff einer heterogenen Nutzerschaft, mit unterschiedlicher Finanzierung bzw. Förderstruktur eine Abrechnung genutzter Grid-Ressourcen zukünftig unumgänglich. Dies setzt einen verbesserten Schutz der Nutzer vor missbräuchlicher Nutzung von Grid-Ressourcen in ihrem Namen und den damit verbundenen Haftungsfragen voraus. Beide Aspekte erfordern den Einsatz von Mechanismen zur beschränkten Delegation, mittels derer Nutzer eine – auch im Nachhinein nachvollziehbare – Freigabe zur Nutzung spezifizierter Ressourcen erteilen können.

Für die Weiterentwicklung des erarbeiteten Ansatzes sowie der Implementierung bieten sich zwei Richtungen an. Naheliegend ist die Unterstützung weiterer Dienste der gLite-Middleware. Zukünftig sollten alle Dienste einer Middleware unterstützt werden, für die Delegationen erteilt werden, so dass die Notwendigkeit zur Delegation unbeschränkter Privilegien durch den Nutzer vollständig entfallen kann. Darüber hinaus ist eine Portierung auf andere Grid-Middlewares anzustreben. Wegen der technischen Kompatibilität und Verbreitung ist hier insbesondere das *Globus Toolkit* zu nennen. Die Unterstützung weiterer Grid-Middlewares ist vor allem in den Grid-Verbünden unerlässlich, in denen mehrere Middlewares parallel eingesetzt werden, wie beispielsweise die D-Grid-Infrastruktur. Hier sollte ein gleichwertiges Sicherheitsniveau für alle Middlewares angestrebt werden, um ein Umgehen von Sicherheitsmechanismen durch Nutzung einer anderen Middleware zu verhindern.

A Die Grid-Testumgebung am RRZN

Parallel zu der Vorbereitung des Projekts D-Grid wurde am Lehrgebiet Rechnernetze des Regionalen Rechenzentrums für Niedersachsen (RRZN) mit dem Aufbau einer umfassenden Grid-Testumgebung begonnen. In dieser sollte unterschiedliche Grid-Software flexibel ausgetestet werden können und so Kompetenz im Bereich des Grid-Computing am RRZN aufgebaut werden. Weiterhin bildet die Testumgebung die Basis für eigene Arbeiten im Bereich der Netzwerksicherheit sowie der Authentifizierungs- und Autorisierungsmechanismen. Innerhalb der Testumgebung werden das *Globus Toolkit* und die gLite-Middleware unterstützt. Die Wahl auf diese beiden Middlewares fiel aufgrund ihrer großen europäischen und weltweiten Verbreitung sowie ihrer Bedeutung für die nationale Infrastruktur.

Die Testumgebung ist in nationale und internationale Aktivitäten eingebunden. Diese waren die Beteiligung am europäischen EGEE-Projekt über eine assoziierte Partnerschaft und Teilnahme an den Aktivitäten zum Testen der gLite-Middleware im Rahmen des Teilprojekts *JRA1-testing*. Weiterhin ist die Testumgebung in die nationalen Infrastrukturen eingebunden, die im Rahmen der D-Grid-Projekte aufgebaut werden.

A.1 Architektur

Bei der Planung der gLite-Testumgebung am RRZN wurde angestrebt, eine vollständige Grid-Infrastruktur abzubilden. Ein Ziel dieser Planungen war es, möglichst geringe Abhängigkeiten von externen Infrastrukturen einzugehen. Damit sollte erreicht werden, dass Funktionalitäten und Dienste der verwendeten gLite-Komponenten mit den höchstmöglichen Freiheitsgraden getestet werden konnten, ohne externe Partner zu beeinträchtigen.

Abbildung A.1 auf der nächsten Seite zeigt die Testumgebung unter Verzicht auf die Darstellung von Komponenten externer Partner, da sie für diese Arbeit nicht relevant sind. Dargestellt sind die Rechnerkomponenten und die für den Betrieb des Grid entscheidenden Kommunikationsbeziehungen zwischen ihnen. Von der Darstellung der Kommunikation ohne Grid-Bezug wie ICMP-Nachrichten oder DNS-Abfragen wurde aus Gründen der Übersichtlichkeit abgesehen.

Obwohl die gLite-Middleware eine parallele Installation von mehreren Diensten auf einem Rechner in der Regel zulässt, wurde bei den meisten Diensten auf diese Methodik bewusst verzichtet. Die separate Installation von einem Dienst je Rechner

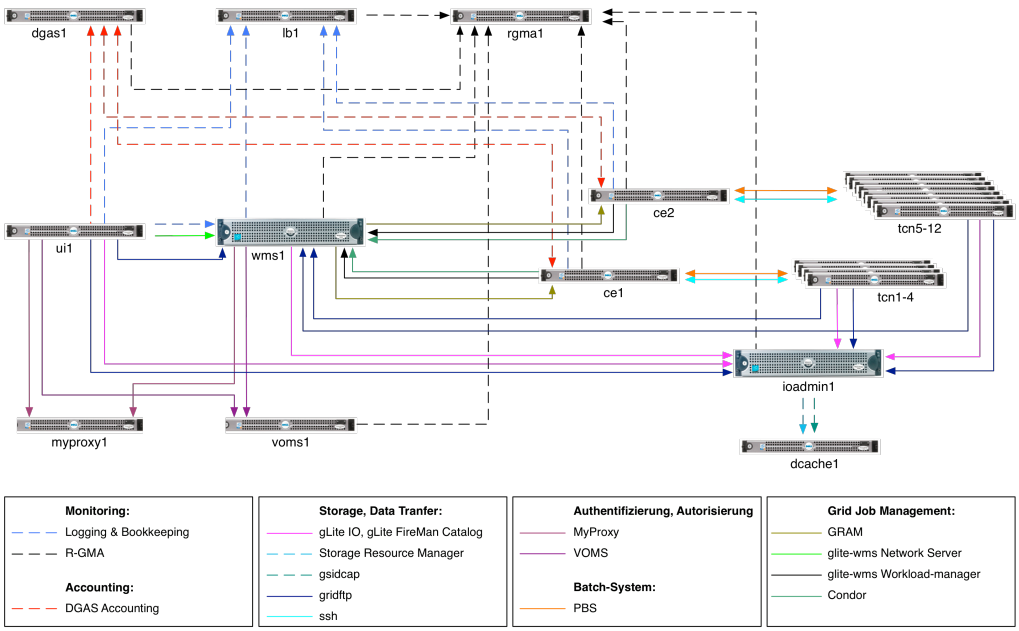


Abbildung A.1: Die gLite-Testumgebung am RRZN

erleichtert sowohl die Fehlersuche als auch die Untersuchung von Kommunikationsbeziehungen erheblich.

Wie aus Abbildung A.1 ersichtlich ist, wurden einige Dienste mehrfach ausgelegt. Dies betrifft die beiden CE-Systeme *ce1* und *ce2* sowie die daran angeschlossenen baugleichen *Worker Nodes tcn1–tcn4* sowie *tcn5–tcn12*. Mit dieser asymmetrischen Verteilung von Rechenleistung sollte die Untersuchung der Funktionalitäten des WMS im Bereich der Lastverteilung ermöglicht werden.

Die Rechnerausstattung der gLite-Testumgebung ist Tabelle A.1 zu entnehmen. Abgesehen von den *Worker Nodes* (Dual AMD Opteron) besteht sie aus Rechnern mit Dual-CPU oder Single-CPU/Dual-Core Intel x86-Prozessoren. Die Rechner sind mit Gigabit-Ethernet an das Netzwerk der Testumgebung angebunden, die WN-Systeme verfügen darüber hinaus über einen Infiniband-Interconnect. Der Arbeitsspeicher dieser Rechner ist mit zwei bzw. vier Gigabyte auch für Java-basierte Grid-Dienste ausreichend dimensioniert. Alle Rechner sind mit redundanten Festplatten ausgestattet, die entweder im RAID-Level 1 oder RAID-Level 5 betrieben werden. Die Ausstattung mit Festplattenspeicher ist lediglich im Fall des SE-Systems *dcache1* relevant, das für die Speicherung von Nutzerdateien zuständig ist. Mit 1,5 TB ist es für den Testbetrieb ausreichend dimensioniert.

Name	Prozessor	RAM	HD	Netzinterface
dgas1, lb1, rgma1, ui1, ce1, ce2, myproxy1, voms1	Intel PentiumD 3 GHz	2 GB	146 GB	1000Base-T
wms1	Dual Intel Xeon 3 GHz	4 GB	300 GB	1000Base-T
ioadmin1	Dual Intel Xeon 2.8 GHz	4 GB	100 GB	1000Base-T
dcache1	Dual Intel Xeon 2.8 GHz	4 GB	146 GB 1,5 TB	1000Base-T
tcn1–12	Dual AMD Opteron 2.2 GHz	4 GB	146 GB	1000Base-T, Infiniband

Tabelle A.1: Rechnerausstattung der gLite-Testumgebung am RRZN

A.2 Dienste

Die in der Testumgebung installierten Dienste der gLite-Middleware umfassen die in Abschnitt 2.1.3 vorgestellten vier Gruppen der *Job Management Serices*, *Data Services*, *Security Services* und *Information and Monitoring Services*. Als Betriebssystem wird auf allen Systemen die Linux-Distribution *Redhat Enterprise Linux Advanced Server 3* oder die binärkompatible Distribution *Scientific Linux 3* verwendet. Der Einsatz dieser schon zum Zeitpunkt des Verfassens dieser Arbeit veralteten Distributionen ist aus Gründen der Kompatibilität mit gLite Version 3.0 notwendig. Ein Übergang auf neuere Versionen wird für zukünftige Versionen von *gLite* jedoch angestrebt [INF07].

Die gLite-Middleware wird durch die von der EU geförderten Projekte EGEE, bzw. den Nachfolger EGEE-II aktiv entwickelt. Dadurch bedingt wurde die gLite-Testumgebung im Laufe ihres Bestehens regelmäßig auf neue Versionen einzelner Komponenten aktualisiert. Die in den Tabellen A.2 bis A.5 angegebenen Versionen beziehen sich auf den zum Zeitpunkt des Verfassens dieser Arbeit aktuellen Stand.

A.2.1 Job Management Serices

Die *Job Management Serices* der gLite-Testumgebung umfassen alle Dienste, die die gLite-Middleware für diesen Bereich vorsieht. Eine Erläuterung der Aufgaben dieser Dienste wurde in Kapitel 2.1.3.1 vorgenommen.

Zusätzlich ist ein Rechner als *User Interface* konfiguriert. Auf diesem haben die Nutzer eigene UNIX-Accounts, auf die sie über einen Secure Shell (SSH) Zugang zugreifen können. Die Aufgabe dieses Systems ist es, dem Nutzer eine Arbeitsum-

Dienst	Rechner	Version	Bemerkungen
User Interface	ui1	glite-UI-3.0.21-0	zusätzlich Zugang zu Globus-Testumgebung
Workload Management System	wms1	glite-WMS-2.3.27-0	
Computing Element	ce1	glite-CE-2.4.32-0	TORQUE Admin Node für tcn1-4
Computing Element	ce2	glite-CE-2.4.32-0	TORQUE Admin Node für tcn5-12
Worker Node	tcn1-4	glite-WN-3.0.13-0	TORQUE Client Nodes für ce1
Worker Node	tcn5-12	glite-WN-3.0.13-0	TORQUE Client Nodes für ce2

Tabelle A.2: Job Management Services in der gLite-Testumgebung

gebung bereitzustellen, um seine Rechenaufträge an das Grid vorzubereiten und sie an das Grid abzugeben. Über die auf diesem System installierten Kommandozeilenbefehle kann der Nutzer mit allen für ihn relevanten Grid-Diensten interagieren.

A.2.2 Data Services

Der Bereich der *Data Services* wird gebildet aus den beiden Diensten *FireMan Data Catalog* und IO-Service, der auf einen SRM-Dienst als SE zurückgreift (vgl. Kapitel 2.1.3.2). Als SRM-Service kommt in der gLite-Testumgebung ein auf *dCache* basierendes Speichersystem zum Einsatz.

Sowohl der *FireMan Data Catalog* als auch der IO-Service werden in zukünftigen Versionen der gLite-Middleware nicht mehr enthalten sein. Sie werden mittelfristig durch den *LCG File Catalog* (LFC) und die *Grid File Access Library* (GFAL) ersetzt, die einen direkten Zugriff des Nutzers auf den SRM-Service ermöglicht. Diese Entscheidung wurde in der Hauptsache mit mangelnder Leistungsfähigkeit der beiden Komponenten begründet. Insbesondere im Bereich der Zugriffskontrolle sind LFC und GFAL jedoch noch ungeeignet, einen Ersatz für die beiden zu ersetzenden Systeme zu bilden, da die verwendeten SE-Systeme noch keine Autorisierung auf Basis der *Access Control Lists* des *File Catalog* sowie der VO-Attribute beherrschen.

Dienst	Rechner	Version	Bemerkungen
FireMan Data Catalog	ioadmin1	glite-data-catalog-service-fr-mysql-1.4.7-2	Service in gLite 3.0 deprecated
IO-Service	ioadmin1	glite-data-io-daemon 2.0.2-2	Service in gLite 3.0 deprecated
Storage Resource Manager	dcache1	dcache-server-1.6.6-5	Service ist keine gLite Komponente

Tabelle A.3: Data Services in der gLite-Testumgebung

A.2.3 Security Services

Die *Security Services* der gLite-Middleware umfassen zwei Dienste aus dem Bereich der Authentifizierung bzw. Autorisierung, den MyProxy sowie den *Virtual Organisation Membership Service*. Die Aufgaben dieser Dienste werden in Kapitel 2.1.3.3 erläutert.

Dienst	Rechner	Version	Bemerkungen
MyProxy Credential Management Service	myproxy1	myproxy-1.14-EGEE	
Virtual Organisation Membership Service	voms1	glite-VOMS_mysql_3.0.13-0	

Tabelle A.4: Security Services in der gLite-Testumgebung

A.2.4 Information and Monitoring Services

Die *Information and Monitoring Services* der gLite-Middleware werden in der Testumgebung mit einem LB-Service sowie einem RGMA-Dienst repräsentiert. Die Aufgaben dieser Dienste sind in Kapitel 2.1.3.4 erläutert.

Zusätzlich zu den genannten Diensten ist in der gLite-Testumgebung noch ein Server für das *Distributed Grid Accounting System* (DGAS) [GPP07] installiert. Diese Software ist aus dem *DataGrid Accounting System* hervorgegangen und implementiert eine verteilte Architektur für das *Accounting* und *Pricing* von Ressourcennut-

Dienst	Rechner	Version	Bemerkungen
Logging & Bookkeeping	lb1	glite-LB-2.2.15-0	
Relational Grid Monitoring Architecture	rgma1	glite-rgma-server-config-5.2.6-0	
Distributed Grid Accounting System	dgas1	glite-dgas-3.1.1	Service in gLite 3.0 nicht enthalten

Tabelle A.5: Information and Monitoring Services in der gLite-Testumgebung

zung. Das DGAS war bis gLite Version 1.5 Bestandteil der Middleware, wird jedoch im Rahmen des EGEE-II Projektes weiterentwickelt.

A.3 Authentifizierung von Nutzern und Diensten

Die Authentifizierung von Nutzern und Diensten in der gLite-Testumgebung erfolgt mittels PKI und X.509-Zertifikaten (vgl. Kapitel 2.2.2.2). Die Dienste der Testumgebung sind derart konfiguriert, dass X.509-Zertifikate von allen Zertifizierungsstellen zur Authentifizierung akzeptiert werden, die Mitglied der *European Grid Policy Management Authority* (EUGridPMA) [EUG07] sind. Zertifizierungsstellen, die Mitglied dieser Organisation sind, müssen nach deren Richtlinien [EUG05] zum Betrieb arbeiten. Das Ziel dieser Vorgaben ist es, ein einheitliches Sicherheitsniveau aller Zertifizierungsstellen und damit eine Vertrauensbeziehung zwischen diesen sowie ihren Nutzern zu etablieren.

Die von den Diensten und lokalen Nutzern der gLite-Testumgebung verwendeten X.509-Zertifikate werden von den beiden deutschen Mitgliedern der EUGridPMA ausgestellt. Diese beiden Zertifizierungsstellen sind die *DFN-Verein PCA Grid*, die vom DFN-Verein betrieben wird, sowie die *GridKa-CA* des Forschungszentrums Karlsruhe (FZK). Beide Zertifizierungsstellen stellen Zertifikate mit Gültigkeitsdauern von 13 Monaten aus.

Der Rückruf von Zertifikaten erfolgt im Falle beider Zertifizierungsstellen über die in Kapitel 2.2.2.3 erläuterten *Certificate Revocation Lists*. Auf allen Rechnern der gLite-Testumgebung werden die Sperrlisten automatisiert alle sechs Stunden aktualisiert. Dieser Zeitraum stellt einen Kompromiss dar zwischen einem möglichst hohen Sicherheitsniveau durch eine schnelle Sperrung kompromittierter Identitäten und dem Ziel einer noch akzeptablen Belastung der Server der Zertifizierungsstellen.

A.4 Heterogene Mechanismen zur Zugriffskontrolle

In gLite-Infrastrukturen wird eine Autorisierung auf Basis des VO-Modells (vgl. Kapitel 2.3.4) angestrebt, in der Instanzen des VOMS als Attributautoritäten fungieren. In der gLite-Testumgebung wird dieses Modell mit dem Betrieb von zwei VOMS-Instanzen umgesetzt, die jeweils eine VO verwalten. Ihre Aufgabe ist es, den Mitgliedern der von ihnen verwalteten VO nach Prüfung der Identität signierte Attributzertifikate [FH02] auszustellen. Diese Attributzertifikate enthalten Daten über die Eigenschaften des Nutzers innerhalb der VO in Form von FQAN-Strings (vgl. Kapitel 2.3.4.1) oder Attribut/Wert-Paaren. Die von den VOMS-Instanzen verwalteten Virtuellen Organisationen sind:

1. VO „rvs“ für lokale Mitarbeiter
2. VO „ext“ für externe Partner

Auf den Grid-Ressourcen wird die Zugriffskontrolle auf Basis der Autorisierungsinformationen innerhalb der Attributzertifikate durchgeführt. Für eine sichere Zugriffskontrolle muss die Ressource verifizieren können, dass das durch den Nutzer präsentierte Attributzertifikat von einer VOMS-Instanz ausgestellt wurde, der sie vertraut. Diese Befähigung wird erreicht, indem jeder Ressource, die Zugriffskontrollen auf Basis der VO-Attribute durchführen soll, das X.509-Zertifikat der ausstellenden VOMS-Instanzen zugänglich gemacht und als vertrauenswürdig markiert wird. Die Ressourcen der Testumgebung unterstützen neben den beiden lokal administrierten Virtuellen Organisationen noch jeweils eine im Rahmen von EGEE (*egtest*) und D-Grid (*dgtest*) verwendete.

Einige Kernkomponenten der gLite-Middleware, wie der LB-Service oder das WMS, beherrschen lediglich Autorisierungsmechanismen der GSI (vgl. Kapitel 2.2.3). Die GSI beherrscht keine Zugriffskontrolle anhand von VO-Attributen, sie entscheidet über die Zugriffe der Nutzer individuell auf Basis des *Distinguished Name* im Subjekt des präsentierten Zertifikats. Zu diesem Zweck existiert auf diesen Komponenten eine Datei, das so genannte *grid-mapfile*, das die *Distinguished Names* der Nutzerzertifikate sowie die Namen der Nutzerkonten enthält, die diese verwenden dürfen. Bei Zugriffen von Nutzern wird jeweils überprüft, ob der DN des zur Authentifizierung genutzten Zertifikats im *grid-mapfile* der Ressource enthalten ist. Ist dies der Fall, wird der Zugriff auf diese gestattet.

Um die Einträge im *grid-mapfile* nicht manuell konfigurieren zu müssen, existiert in der gLite-Middleware ein Mechanismus, diese automatisiert mit den Nutzerlisten der in der jeweiligen Infrastruktur unterstützten VO bzw. deren VOMS-Instanzen abzugleichen. Auch wenn dieses Verfahren einen der Nachteile der Autorisierung anhand des grid-mapfile abmildert, bleiben doch gravierende Nachteile. So ist es mit diesem Verfahren unmöglich, mehrere Rollen eines Nutzers innerhalb der VO zu differenzieren. Weiterhin können bei simultaner Mitgliedschaft in mehreren Virtuellen Organisationen – abhängig von der jeweils aktiven – keine unterschiedlichen

Betriebsmittel innerhalb der Ressource zugewiesen werden. Mittelfristig wird daher angestrebt, vollständig auf VO- und damit RBAC-basierte Autorisierungsmodelle (vgl. Kapitel 2.3.1.3) überzugehen.

B Abgabe von Rechenaufträgen mittels Web-Portal

In Kapitel 5.3.2 wurde die grafische Nutzerschnittstelle sowie die für diese implementierten Komponenten vorgestellt. Neben den grundlegenden Funktionalitäten für die Spezifikation und Abgabe von Rechenaufträgen stellt die Nutzerschnittstelle Werkzeuge für die intuitive Erstellung von nutzerdefinierten Richtlinien bereit, mittels derer der Zugriff auf Objekte der SE-Systeme limitiert werden kann.

Die bereitgestellten Funktionalitäten sollen in diesem Abschnitt anhand eines exemplarischen Rechenauftrags verdeutlicht werden. Im Rahmen des Rechenauftrags wird ein Skript für die Bash-Shell ausgeführt, das während der Abarbeitung über den IO-Service auf Dateien eines SE-Systems zugreift.

B.1 Nutzung von Proxy-Credentials innerhalb des Web-Portals

Nach der Anmeldung am *Gridsphere*-Portal, auf die hier nicht weiter eingegangen wird, hat der Nutzer die Auswahl zwischen den drei Reitern „Credentials“, „Resources“ und „Jobs“. Um dem Portal die Interaktion mit Grid-Komponenten in seinem Namen zu ermöglichen, muss der Nutzer ein Proxy-Credential auf das Portal transferieren. Dieser Vorgang erfolgt im Reiter „Credentials“. In Abbildung B.1 auf der nächsten Seite ist der Reiter dargestellt, nachdem ein Proxy-Credential durch den Nutzer auf das Web-Portal transferiert wurde. Dargestellt werden unter „Verfügbare Credentials“ das Subjekt des Proxy-Zertifikats sowie dessen Restlaufzeit. Um dem Nutzer einen schnellen Überblick über die Verfügbarkeit eines gültigen Proxy-Credentials zu geben, wird weiterhin der aktuelle Status angegeben. In diesem Fall ist das Proxy-Credential als „Valid“ gekennzeichnet, es kann folglich für Aktionen im Grid verwendet werden.

Dem Nutzer bieten sich zwei Optionen, dem Web-Portal Proxy-Credentials verfügbar zu machen. Zum Einen kann er ein auf dem UI erstelltes Proxy-Credential auf das Portal hochladen. Dies erfolgt über die Auswahl der lokalen Datei, in der sich das Proxy-Credential befindet, mittels „Choose File“ (vgl. Abbildung B.1 auf der nächsten Seite) und einer nachfolgenden Betätigung des Schalters „Credential hochladen“. Zum Anderen kann er eine Delegation von einem MyProxy-Dienst anfordern, dem er zuvor eine Delegation erteilt hat. Dieser Vorgang erfolgt im „MyProxy Portlet“ unter Angabe des Namens der Delegation sowie dem Benutzernamen und verwendeten Passwort. Nach Auswahl des MyProxy-Servers muss noch die Virtuelle Organisation angegeben werden, von deren VOMS-Server ein Attri-

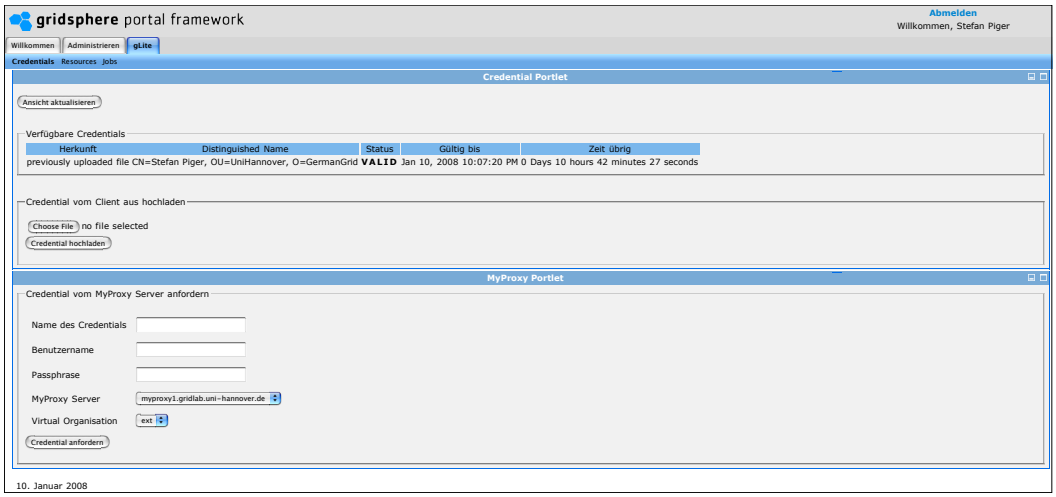


Abbildung B.1: Verwaltung von Proxy-Credentials

butzertifikat zur Einbettung in das Proxy-Credential angefordert wird. Nach dem erfolgreichen Abruf der Proxy-Credentials vom MyProxy-Server muss der Nutzer noch die gewünschte Rolle innerhalb der VO aus den ihm zugeordneten auswählen (nicht dargestellt). Diese werden als Attributzertifikat vom VOMS-Server der VO bezogen und in ein weiteres Proxy-Zertifikat eingebettet. Nachdem der Nutzer eine der beiden Optionen gewählt und die entsprechenden Aktionen ausgeführt hat, steht dem Web-Portal ein gültiges Proxy-Credential zur Verfügung. Der Nutzer kann nun einen Rechenauftrag initiieren. Dazu ist der Reiter „Jobs“ auszuwählen.

B.2 Spezifikation und Beauftragung des Jobs

Das *Portlet* „Jobs“ ist aus Gründen der Übersichtlichkeit in mehrere Ansichten unterteilt. Der Nutzer kann zwischen diesen mittels einer Navigationsleiste (vgl. z. B. Abbildung B.3 auf Seite 141) wechseln. Dadurch bietet sich dem Nutzer die Möglichkeit, innerhalb des *Workflows* zur Beauftragung von Rechenjobs Änderungen an der Konfiguration vorzunehmen, die er in einer vorangehenden Ansicht vorgenommen hat. Das *Portlet* ist zustandsbehaftet, d. h. die Informationen des Nutzers gehen durch das Wechseln in eine andere Ansicht nicht verloren.

B.2.1 Verwaltung von Rechenaufträgen

Der Reiter „Jobs“ bietet dem Nutzer in der ersten Ansicht „Job list“ (vgl. Abbildung B.2 auf der nächsten Seite) eine Übersicht über seine Rechenaufträge. Diese

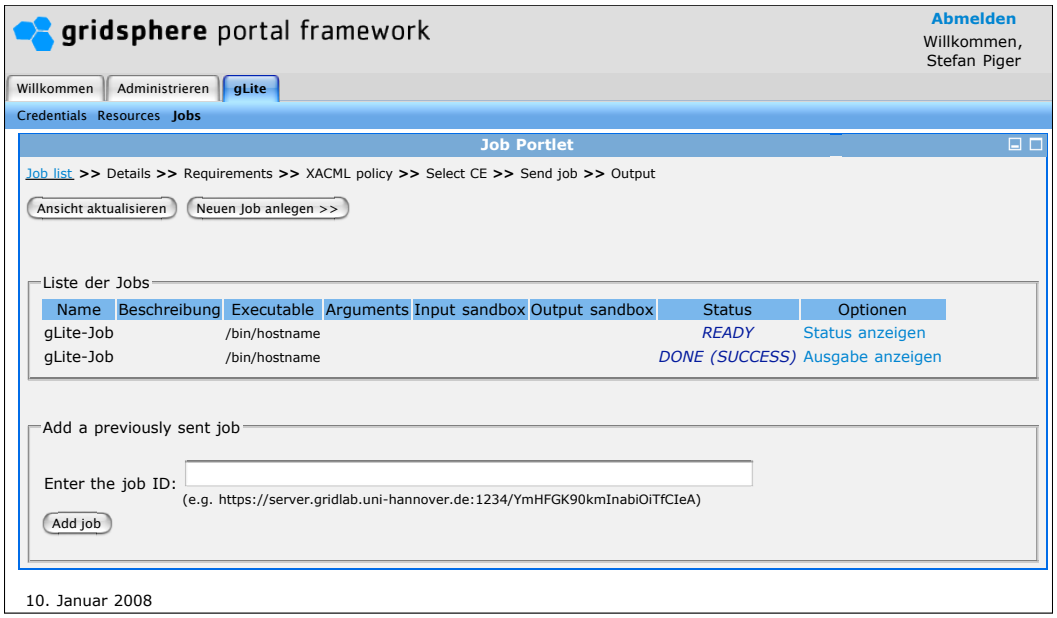


Abbildung B.2: Übersicht über Rechenaufträge

bietet neben dem Namen des Auftrags sowie einer optionalen Beschreibung Informationen über das verwendete Programm, den Argumenten, mit denen es gestartet wurde, und den in den *Input-* und *Output-Sandboxes* übertragenen Dateien. Weiterhin wird der aktuelle Status der Aufträge angezeigt und eine Option geboten, Details über deren Status zu erhalten bzw. bei abgeschlossenen Aufträgen die Ausgabedateien der *Output-Sandbox* vom zuständigen WMS abzurufen.

Das Web-Portal fügt der Liste der Rechenaufträge automatisch alle über dieses abgegebene Aufträge hinzu. Der Nutzer kann jedoch manuell Aufträge hinzufügen, die er über die Kommandozeile des UI abgegeben hat. Dazu muss er deren *JobID* in die Eingabezeile der Ansicht „Job list“ einfügen und den Schalter „Add job“ betätigen.

Das Anlegen neuer Rechenaufträge erfolgt in der Ansicht „Job list“ (vgl. Abbildung B.2) durch Betätigung des Schalters „Neuen Auftrag anlegen >>“. Nachfolgend wechselt das Web-Portal in die Ansicht „Details“.

B.2.2 Anlegen eines Rechenauftrags

Die Ansicht „Details“ (vgl. Abbildung B.3 auf Seite 141) innerhalb des *Portlets* „Jobs“ bietet dem Nutzer Optionen zur Spezifikation eines Rechenauftrags vom Typ *Normal*. Es ist in fünf Gruppen gegliedert, wobei die letzte Gruppe lediglich

die erstellte Job-Spezifikation in JDL-Syntax zu Kontrollzwecken ausgibt. In der ersten Gruppe, die sich unterhalb der Navigationsleiste befindet, muss der Nutzer dem Auftrag einen Namen geben, über den dieser in der „Job list“ referenziert wird. Optional kann noch eine Beschreibung hinzugefügt werden. Auch können Namen für die Standardausgabe und -fehlerdatei vergeben werden.

Die zweite Gruppe gibt dem Nutzer die Möglichkeit, eine ausführbare Datei zu spezifizieren. Dafür kann entweder der Name und Pfad eines auf dem WN vorliegenden Programms spezifiziert werden oder ein Programm auf das Portal transferiert werden, das automatisch der Input-Sandbox des Auftrags hinzugefügt wird. Weiterhin können optional Argumente angegeben werden, mit denen die ausführbare Datei gestartet wird.

In der dritten Gruppe können weitere Dateien für die Input-Sandbox spezifiziert und auf das Web-Portal transferiert werden. Die in Abbildung B.3 auf der nächsten Seite dargestellte Input-Sandbox enthält bereits die Datei „mydatatest.sh“. Diese wurde durch ihre Spezifikation als ausführbare Datei automatisch durch das Web-Portal hinzugefügt.

Die vierte Gruppe dient der Spezifikation von weiteren Dateien für die Output-Sandbox. In den Grundeinstellungen enthält diese bereits die Standardausgabe und -fehlerdatei. Erzeugt der Rechenauftrag weitere Ausgabedateien, die nicht während dessen Abarbeitung auf ein SE transferiert werden, können diese mittels der Output-Sandbox bei Abschluss des Auftrags auf das WMS transferiert und von dort auf das Web-Portal übertragen werden.

Nach Abschluss der grundlegenden Konfiguration muss der Schalter „Anforderungen (Requirements) auswählen >>“ betätigt werden, um in die Ansicht „Requirements“ zu wechseln.

B.2.3 Spezifikation von Anforderungen an das ausführende Computing Element

Die Ansicht „Requirements“ (vgl. Abbildung B.4 auf Seite 142) dient der Spezifikation von Anforderungen an das ausführende CE. Die zur Auswahl stehenden Kriterien sind über Auswahlmenüs erreichbar und können über einen Operator mit einem Wert verknüpft werden. Vorkonfiguriert sind folgende Kriterien:

- **other.GlueHostMainMemoryRAMSize.** Mit diesem Kriterium lassen sich Minimalanforderungen bezüglich des auf den *Worker Nodes* zur Verfügung stehenden Arbeitsspeichers spezifizieren.
- **other.GlueCEStateFreeCPUs.** Option zur Spezifikation einer Anforderung bezüglich der Anzahl von freien CPUs, die durch das CE verwaltet werden.

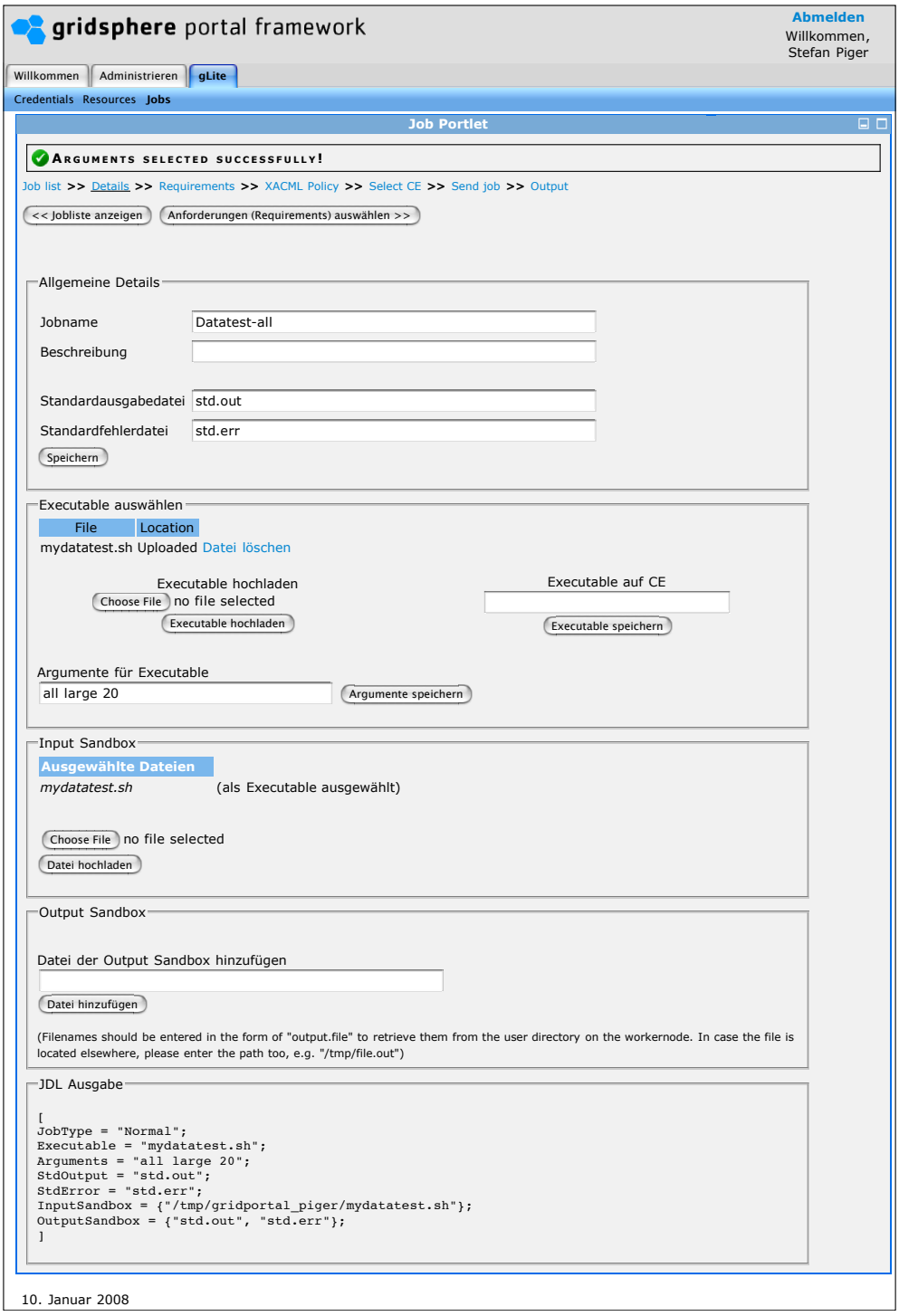


Abbildung B.3: Spezifikation der Basisparameter eines Auftrags

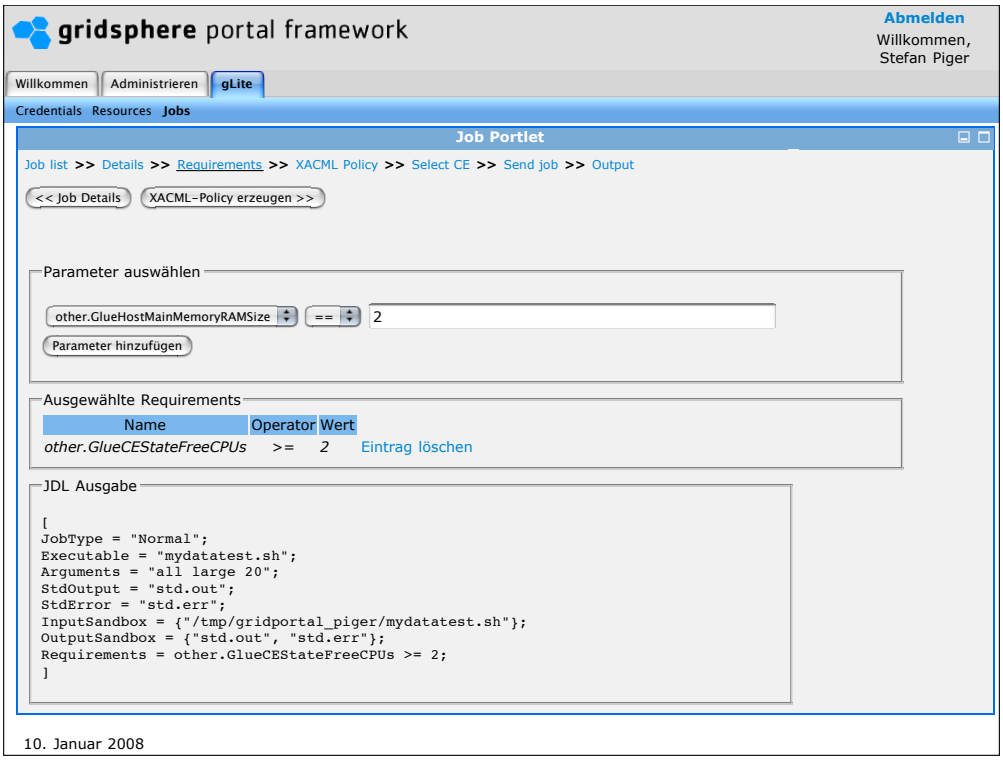


Abbildung B.4: Spezifikation der Anforderungen an die ausführende Infrastruktur

- **other.GlueCEStateTotalCPUs.** Option zur Spezifikation einer Anforderung bezüglich der Anzahl von CPUs, die durch das CE verwaltet werden.
- **other.GlueCEPolicyMaxCPUTime.** Option zur Spezifikation einer Anforderung bezüglich der maximalen CPU-Zeit, die einem Auftrag zur Verfügung steht.
- **other.GlueCEStateFreeJobSlots.** Option zur Spezifikation einer Anforderung bezüglich der momentan freien Plätze für Rechenaufträge.
- **other.GlueCEStateStatus.** Option zur Spezifikation einer Anforderung bezüglich des Zustands eines CE. Damit lassen sich z. B. nur *Computing Elements* für die Bearbeitung eines Auftrags auswählen, die im Zustand *Production* sind und somit eine gewisse Verfügbarkeit garantieren.

Als Operatoren stehen „==“, „>=“, „<=“, „>“ und „<“ zur Verfügung, die Werte können durch den Nutzer frei spezifiziert werden. Nach Abschluss der Konfiguration der Anforderungen muss der Schalter „XACML-Policy erzeugen >>“ betätigt werden, um in die Ansicht „XACML Policy“ zu wechseln.

B.2.4 Erstellen der nutzerdefinierten Richtlinie

Die Ansicht „XACML Policy“ (vgl. Abbildung B.5 auf der nächsten Seite) bietet dem Nutzer in vier Gruppen Funktionalitäten zur Spezifikation von nutzerdefinierten Richtlinien für den Zugriff auf Dateien und Verzeichnisse des *File Catalog* sowie verschiedene Übersichten der gewählten Regeln.

In der ersten Gruppe unterhalb der Navigationsleiste kann der Nutzer innerhalb der Verzeichnisstruktur des *File Catalog* navigieren und für die dort vorhandenen Objekte Regeln für den lesenden, schreibenden, erstellenden und löschenden Zugriff durch Anwählen der entsprechenden Boxen festlegen. Weiterhin ist es möglich, Regeln für das jeweilig aktuelle Verzeichnis zu spezifizieren. Eine Regel wird durch Betätigung des Schalters „Regel hinzufügen“ in der entsprechenden Zeile der nutzerdefinierten Richtlinie hinzugefügt.

In der zweiten Gruppe können Zielobjekte in Form von *Logical File Names* direkt durch den Nutzer angegeben sowie die gewünschten Zugriffsrechte ausgewählt werden. Durch Betätigung des Schalters „Regel hinzufügen“ wird diese der nutzerdefinierten Richtlinie hinzugefügt.

Die dritte Gruppe stellt die spezifizierten Regeln in einer für den Nutzer übersichtlichen Form dar. Hier können auch unerwünschte oder fehlerhaft spezifizierte Regeln wieder entfernt werden.

Die vierte Gruppe gibt erfahrenen Nutzer eine direkte Ansicht der erstellten Richtlinie. Die in Abbildung B.5 auf der nächsten Seite dargestellte Richtlinie stellt einen Ausschnitt der spezifizierten Richtlinie dar.

Durch Betätigung des Schalters „Mögliches Computing Element auswählen >>“, der sich unterhalb der Navigationsleiste befindet, wird in die Ansicht „Select CE“ gewechselt.

B.2.5 Auswahl eines Computing Elements

Die Ansicht „Select CE“ bietet dem Nutzer die Option, ein *Computing Element* für die Ausführung seines Auftrags auszuwählen. Dabei werden dem Nutzer diejenigen Systeme angezeigt, die für die Ausführung des Auftrags als geeignet erscheinen. Die entsprechenden Informationen erhält das Web-Portal vor Anzeige dieser Ansicht durch eine entsprechende Anfrage beim WMS. Durch Betätigen des Schalters „Job abschicken >>“ wird in die Ansicht „Send Job“ gewechselt.

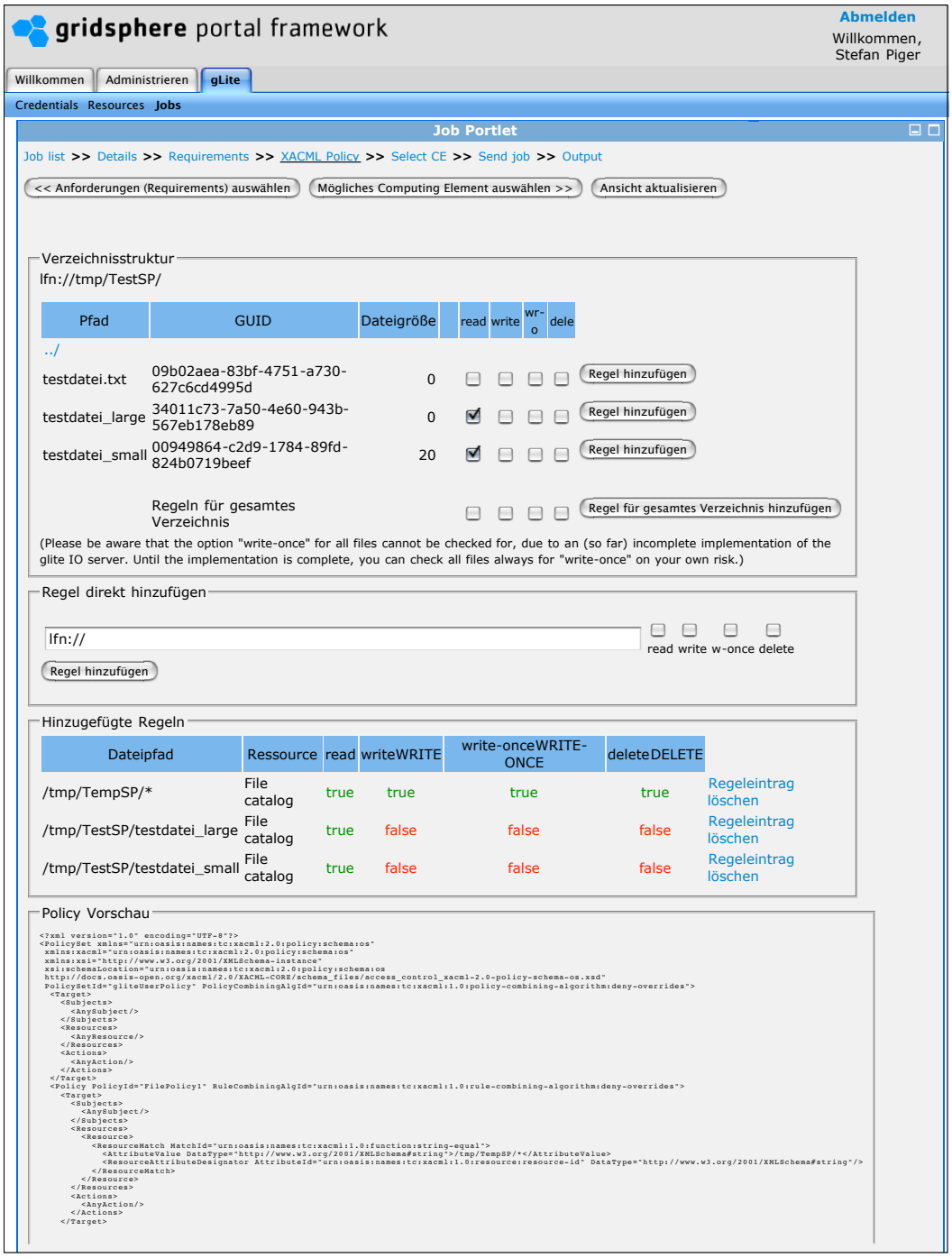


Abbildung B.5: Spezifikation der nutzerdefinierten Richtlinien (Ausschnitt)

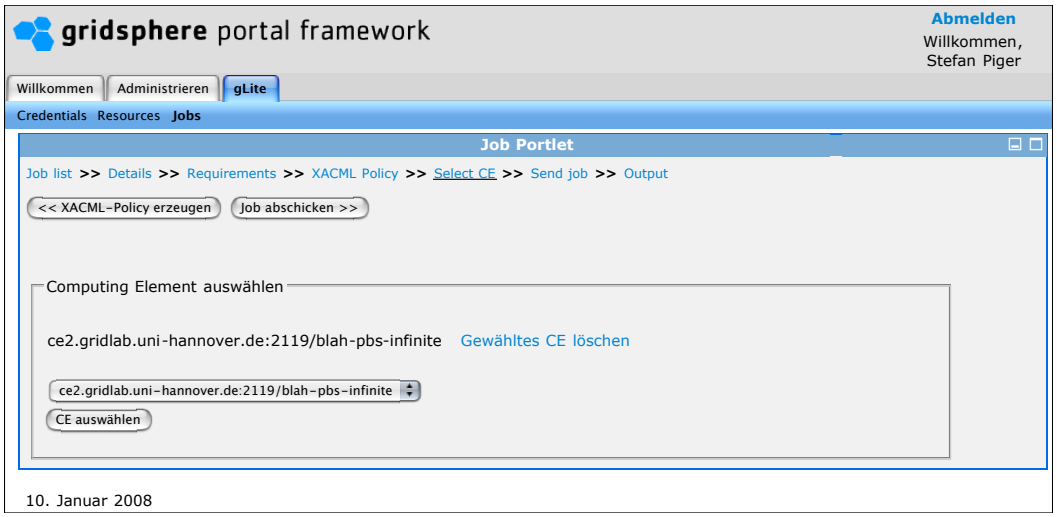


Abbildung B.6: Auswahl eines CE zur Ausführung eines Auftrags

B.2.6 Abgabe des Rechenauftrags

Die Ansicht „Send Job“ bietet dem Nutzer eine Übersicht über die gewählten Parameter sowie einen Schalter („Job abschicken“, nicht dargestellt), mit dem er den Auftrag an das WMS übergeben kann. In Abbildung B.7 auf der nächsten Seite ist der Zustand dieser Ansicht dargestellt, nachdem der Auftrag an das Grid übergeben wurde. Diese Ansicht bietet dem Nutzer zusätzlich Informationen über den Status des Auftrags. Der Detailgrad der Ausgabe ist über das Menü „Ausgabe Details“ in drei Stufen konfigurierbar.

Bei Abgabe des Auftrags an das WMS führt das Web-Portal zwei Ableitungen von Proxy-Zertifikaten durch. Das erste Proxy-Zertifikat wird von dem unbeschränkten Proxy-Zertifikat (vgl. Kapitel B.1) erstellt, das für die Aktionen im Grid während der Spezifikation des Auftrags verwendet wurde. Es enthält eine eingebettete *FilePolicy* mit den durch den Nutzer spezifizierten Regeln. Die zweite Ableitung erfolgt von dem bereits durch die *FilePolicy* beschränkten Proxy-Zertifikat und beinhaltet die *ExecutionPolicy*, die die durch das Web-Portal registrierte *JobID* enthält.

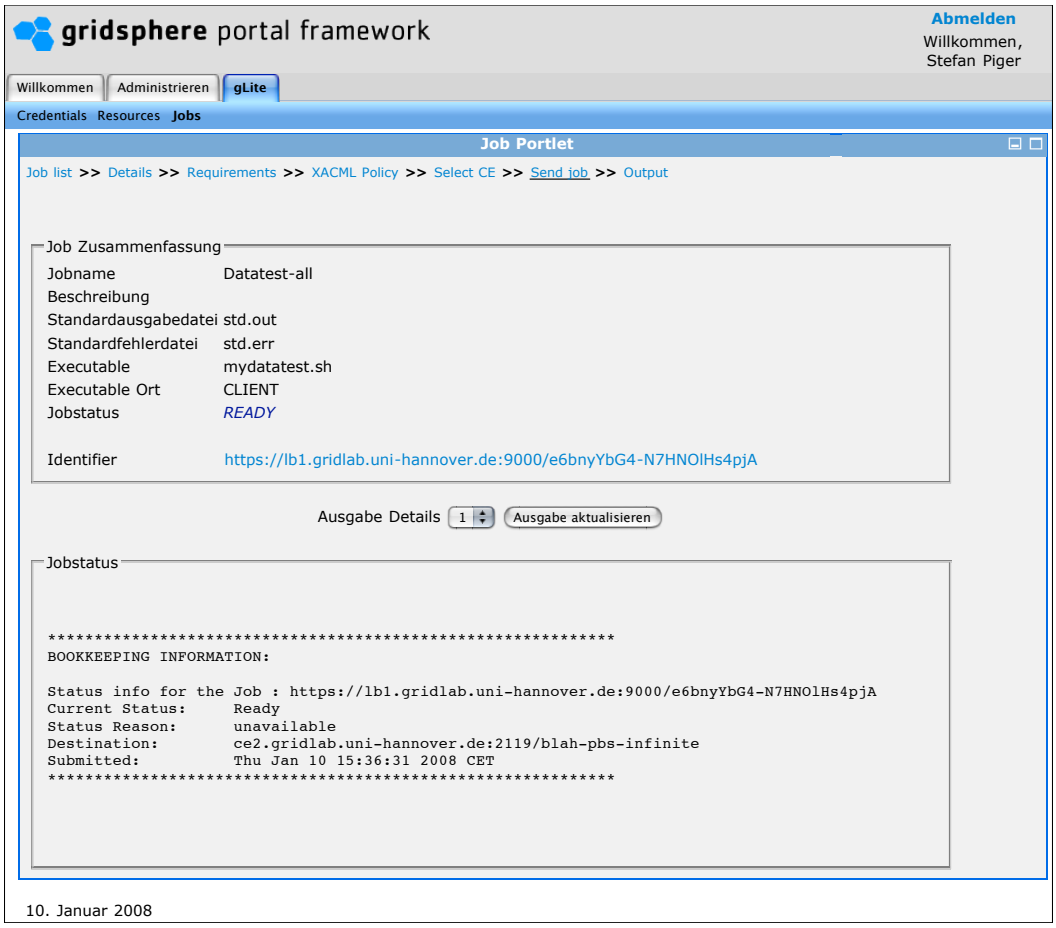


Abbildung B.7: Ansicht nach Abgabe eines Auftrags

Literaturverzeichnis

[ABB+03] Allcock, W., J. Bester, J. Bresnahan, S. Meder und S. Tuecke: *GridFTP: Protocol Extensions to FTP for the Grid*. Global Grid Forum Document GFD.20, 2003.

[ABM+06] Ahsant, M., J. Basney, O. Mulmo, A. J. Lee und L. Johnsson: *Toward An On-demand Restricted Delegation Mechanism for Grids*. In: *Proceedings of the 7th IEEE/ACM International Conference on Grid Computing (Grid2006)*, 2006.

[Apa07] Apache Software Foundation: *Apache Tomcat Homepage*. [Online]. `http://tomcat.apache.org/`, Dezember 2007.

[Bis02] Bishop, M. A.: *The Art and Science of Computer Security*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[BL76] Bell, D. E. und L. J. LaPadula: *Secure Computer System: Unified Exposition and Multics Interpretation*. Technischer Bericht MTR-2997, The MITRE Corporation, März 1976.

[CER07] CERN: *CASTOR – CERN Advanced STORage manager*. [Online]. `http://castor.web.cern.ch/castor/`, 2007.

[Cia05] Ciaschini, V.: *EGEE User's Guide – VOMS Core Services*. [Online]. `https://edms.cern.ch/file/571991/1/voms-guide.pdf`, 2005.

[Clu07] Cluster Resources, Inc.: *TORQUE Resource Manager*. [Online]. `http://www.clusterresources.compages/products/torque-resource-manager.php`, July 2007.

[CO02] Chadwick, D.W. und A. Otenko: *The PERMIS X.509 Role Based Privilege Management Infrastructure*. In: *Proc 7th ACM Symposium On Access Control Models And Technologies (SACMAT 2002), Monterey, USA*, Seiten 135–140, June 2002.

[Con07] Condor Project: *Condor Project Homepage*. [Online]. `http://www.cs.wisc.edu/condor/`, July 2007.

[DA99] Dierks, T. und C. Allen: *The TLS Protocol Version 1.0*, 1999.

[dCa07] dCache.org: *dCache – main page*. [Online]. `http://www.dcache.org`, 2007.

[DCS+05] Di Meglio, A., L. Cornwall, M. Steenbakkers, J. Flammer, R. Harakaly, A. Aimar, D. Collados Polidura und S. Fisher: *Developer's Guide – for the gLite EGEE*

Middleware. [Online]. `https://edms.cern.ch/file/468700/0.7/EGEE-JRA1-TEC-468700-Developers_Guide-v0-7.pdf`, 2005.

[DEI08] DEISA: *DEISA - Distributed European Infrastructure for Supercomputing Applications*. [Online]. `http://www.deisa.org/`, 2008.

[DFP+96] DeFanti, T. A., I. Foster, M. E. Papka, R. Stevens und T. Kuhfuss: *Overview of the I-WAY: Wide-Area Visual Supercomputing*. The International Journal of Supercomputer Applications and High Performance Computing, 10(2/3):123–131, Summer/Fall 1996.

[DH76a] Diffie, W. und M. E. Hellman: *Multiuser Cryptographic Techniques*. In: *Proc. AFIPS 1976 National Computer Conference*, Seiten 109–112, 1976.

[DH76b] Diffie, W. und M. E. Hellman: *New Directions in Cryptography*. IEEE Transactions on Information Theory, IT-22(6):644–654, 1976.

[DOD85] Department of Defense: *Trusted Computer System Evaluation Criteria*, DOD 5200.28-STD, December 1985.

[EGE05] EGEE Security JRA3: *Global Security Architecture for Web and Legacy Services*. [Online]. `https://edms.cern.ch/document/602183/`, September 2005.

[EGE07] EGEE: *Enabling Grids for E-sciencE. Home Page*. [Online]. `http://www.eu-egee.org/`, 2007.

[EUG05] EUGridPMA: *Guidelines and Authentication Profiles: Classic X.509 CAs with secured infrastructure*. [Online]. `http://www.eugridpma.org/guidelines/IGTF-AP-classic-4-1.pdf`, October 2005.

[EUG07] EUGridPMA: *European Policy Management Authority for Grid Authentication*. [Online]. `http://eugridpma.org/`, June 2007.

[FC04] Frohner, Á. und V. Ciaschini: *VOMS Credential Format*. [Online]. `http://edg-wp2.web.cern.ch/edg-wp2/security/voms/edg-voms-credential.pdf`, 2004.

[FCK95] Ferraiolo, D. F., J. A. Cugini und D. R. Kuhn: *Role-Based Access Controls (RBAC): Features and Motivations*. In: *11th Annual Computer Security Applications Proceedings*, 1995.

[FH02] Farrell, S. und R. Housley: *RFC 3281: An Internet Attribute Certificate Profile for Authorization*. [Online]. `http://www.ietf.org/rfc/rfc3281.txt`, April 2002.

[FK92] Ferraiolo, D. F. und D. R. Kuhn: *Role-Based Access Controls*. In: *15th NIST-NCSC National Computer Security Conference*, Seiten 554–563, 1992.

[FK97] Foster, I. und C. Kesselman: *Globus: A Metacomputing Infrastructure Toolkit*. The International Journal of Supercomputer Applications and High Performance Computing, 11(2):115–128, Summer 1997.

[FK99] Foster, I. und C. Kesselman: *The Globus toolkit.* In: *The Grid: Blueprint for a New Computing Infrastructure*, Seiten 259–278. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[FK03] Foster, I. und C. Kesselman: *Concepts and Architecture.* In: *The Grid 2: Blueprint for a New Computing Infrastructure*, Seiten 37–63. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[FKT01] Foster, I., C. Kesselman und S. Tuecke: *The Anatomy of the Grid: Enabling Scalable Virtual Organizations.* In: *Euro-Par '01: Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing*, Seiten 1–4, London, UK, 2001. Springer-Verlag.

[FKTT98] Foster, I., C. Kesselman, G. Tsudik und S. Tuecke: *A Security Architecture for Computational Grids.* In: *Proceedings of the 5th ACM Conference on Computer and Communications Security*, Seiten 83–91, New York, NY, 1998. ACM Press.

[Fos05] Foster, I. T.: *Globus Toolkit Version 4: Software for Service-Oriented Systems.* In: Jin, Hai, Daniel A. Reed und Wenbin Jiang (Herausgeber): *NPC*, Band 3779 der Reihe *Lecture Notes in Computer Science*, Seiten 2–13. Springer, 2005.

[Fos07] Foster, I.: *A Globus Primer.* [Online]. `http://www.globus.org/toolkit/docs/4.0/key/GT4_Primer_0.6.pdf`, 2007.

[FP05] Feichtinger, D. und A. J. Peters: *Authorization of Data Access in Distributed Storage Systems.* In: *Grid Computing, 2005. The 6th IEEE/ACM International Workshop on Grid Computing*, Seiten 172–178, 2005.

[FSG+01] Ferraiolo, D. F., R. Sandhu, S. Gavrila, D. R. Kuhn und R. Chandramouli: *Proposed NIST standard for role-based access control.* ACM Trans. Inf. Syst. Secur., 4(3):224–274, 2001.

[GGKL89] Gasser, M., A. Goldstein, C. Kaufman und B. Lampson: *The Digital Distributed System Security Architecture.* In: *Proc. 12th NIST-NCSC National Computer Security Conference*, Seiten 305–319, 1989.

[GGPW07] Groeper, R., C. Grimm, S. Piger und J. Wiebelitz: *An Architecture for Authorization in Grids using Shibboleth and VOMS.* In: *EUROMICRO '07: Proceedings of the 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO 2007)*, Seiten 367–374, Washington, DC, USA, 2007. IEEE Computer Society.

[Glo06] Globus Alliance: *CAS - Community Authorization Service.* [Online]. `http://www.globus.org/grid_software/security/cas.php`, 2006.

[GM90] Gasser, M. und E. McDermott: *An Architecture for Practical Delegation in a Distributed System.* In: *IEEE Symposium on Security and Privacy*, Band 00, Seiten 20–30, Los Alamitos, CA, USA, 1990. IEEE Computer Society.

[GPP07] Guarise, A., G. Patania und R. M. Piro: *The Distributed Grid Accounting System (DGAS).* [Online]. `http://www.to.infn.it/grid/accounting/main.html`, June 2007.

[Gra07] Grandi, C.: *Security and Job Management.* [Online]. `http://www.ogf.org/OGF20/materials/792/OGF-gLite-070509-v2.ppt`, Mai 2007.

[Gri08] GridPP: *The Grid Security Vulnerability Group (GSVG). Home Page.* [Online]. `http://www.gridpp.ac.uk/gsvg/advisories/`, 2008.

[Gro05] Groep, D.: *Profile for Traditional X.509 Public Key Certification Authorities with secured infrastructure.* [Online]. `http://www.gridpma.org/docs/IGTF-AP-classic-20050828-4-01.pdf`, 2005.

[Gro06] Groeper, R.: *Policy-based Authorization for Grid Data-Management.* Masterarbeit, Gottfried Wilhelm Leibniz Universität Hannover, 2006.

[HLR06] HLRN: *Norddeutscher Verbund für Hoch- und Höchstleistungsrechnen. Home Page.* [Online]. `http://www.hlrn.de/`, 2006.

[HPFS02] Housley, R., W. Polk, W. Ford und D. Solo: *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile.* RFC 3280, April 2002.

[INF05] INFN: *Virtual Organisation Membership Service.* [Online]. `http://infnforge.cnaf.infn.it/voms/`, May 2005.

[INF07] INFN: *EGEE-II: Update of Functional Description of Grid Components and Associated Workplan.* [Online]. `https://edms.cern.ch/document/816743/`, April 2007.

[Int97] International Telecommunication Union: *Recommendation X.509. Information Technology - Open Systems Interconnection - The Directory: Authentication Framework*, August 1997. Obsoleted by [Int05].

[Int02] International Telecommunication Union: *Recommendation X.680 (2002). Information Technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation*, 2002.

[Int05] International Telecommunication Union: *Recommendation X.509 (2005). Information Technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks*, August 2005. Obsoletes [Int97].

[KF98] Kesselman, C. und I. Foster: *The Grid: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann Publishers, November 1998.

[KFJ03] Kagal, L., T. W. Finin und A. Joshi: *A Policy Based Approach to Security for the Semantic Web.* In: *International Semantic Web Conference*, Seiten 402–418, 2003.

[Kle03] Klein, T.: *Buffer Overflows und Format-String-Schwachstellen. Funktionsweisen, Exploits und Gegenmaßnahmen.* Dpunkt Verlag, 2003.

[Kun07] Kunz, C.: *Implementation of a policy-based authorization for gLite Compute Services.* Masterarbeit, Gottfried Wilhelm Leibniz Universität Hannover, 2007.

[KW02] Keahey, K. und V. Welch: *Fine-Grain Authorization for Resource Management in the Grid Environment.* In: *GRID '02: Proceedings of the Third International Workshop on Grid Computing*, Seiten 199–206, London, UK, 2002. Springer-Verlag.

[LAK+03] Lorch, M., D. B. Adams, D. Kafura, M. S. R. Koneni, A. Rathi und S. Shah: *The PRIMA System for Privilege Management, Authorization and Enforcement in Grid Environments.* In: *GRID '03: Proceedings of the Fourth International Workshop on Grid Computing*, Seite 109, Washington, DC, USA, 2003. IEEE Computer Society.

[Lau05] Lauesen, S.: *User Interface Design: A Software Engineering Perspective.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[LHC06] LHC Computing Grid Project: *LHC Computing Grid.* [Online]. `http://lcg.web.cern.ch/lcg/`, 2006.

[Li00] Li, N.: *Delegation Logic: A Logic-based Approach to Distributed Authorization.* Doktorarbeit, New York University, September 2000.

[Lin00] Linn, J.: *Generic Security Service Application Program Interface Version 2, Update 1.* RFC 2743 (Proposed Standard), Januar 2000.

[Lor04] Lorch, M.: *PRIMA Privilege Management and Authorization in Grid Computing Environments.* Doktorarbeit, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, USA, April 2004.

[MAM+99] Myers, M., R. Ankney, A. Malpani, S. Galperin und C. Adams: *RFC2560: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP.* [Online]. `http://www.ietf.org/rfc/rfc2560.txt`, June 1999.

[MWTE04] Meder, S., V. Welch, S. Tuecke und D. Engert: *GSS-API Extensions.* Grid Security Infrastructure (GSI) Working Group of the Global Grid Forum, February 2001, Revised June 2004.

[Neu93] Neuman, B. C.: *Proxy-Based Authorization and Accounting for Distributed Systems.* In: *International Conference on Distributed Computing Systems*, Seiten 283–291, 1993.

[NRW04] Novotny, J., M. Russell und O. Wehrens: *GridSphere: a portal framework for building collaborations: Research Articles*. Concurr. Comput. : Pract. Exper., 16(5):503–513, 2004.

[NTW01] Novotny, J., S. Tuecke und V. Welch: *An Online Credential Repository for the Grid: MyProxy*. In: *HPDC '01: Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10'01)*, Seite 104, Washington, DC, USA, 2001. IEEE Computer Society.

[Org05] Organisation for the Advancement of Structured Information Standards (OASIS): *eXtensible Access Control Markup Language (XACML) Version 2.0*. [Online]. `http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf`, Februar 2005.

[Org07a] Organisation for the Advancement of Structured Information Standards (OASIS): *Language (XACML) Version 2.0 Policy Schema*. [Online]. `http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-policy-schema-os.xsd`, Dezember 2007.

[Org07b] Organisation for the Advancement of Structured Information Standards (OASIS): *Security Assertion Markup Language (SAML)*. [Online]. `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security`, Dezember 2007.

[Pac05] Pacini, F.: *WMS Service User's Guide*. [Online]. `https://edms.cern.ch/document/572489/1`, 2005.

[Pac06] Pacini, F.: *Job Description Language Attributes Specification For The GLite Middleware*. [Online]. `https://edms.cern.ch/file/555796/1/EGEE-JRA1-TEC-555796-JDL-Attributes-v0-8.pdf`, January 2006.

[PGGK08] Piger, S., C. Grimm, R. Groeper und C. Kunz: *A Comprehensive Approach to Self-Restricted Delegation of Rights in Grids*. In: *Proceedings of the 8th IEEE International Symposium on Cluster Computing and the Grid (CCGrid2008)*, 2008.

[PGWG06] Piger, S., C. Grimm, J. Wiebelitz und R. Groeper: *An Approach to Restricted Delegation of User Rights based on the gLite Middleware*. In: *Proc. Cracow Grid Workshop 06*, 2006.

[PKGG08] Piger, S., C. Kunz, C. Grimm und R. Groeper: *Enhancing Security in Grids through Self-Restricted Delegation of Rights with User-based Policies*. In: *Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Networks (PDCN2008)*, 2008.

[Pos80] Postel, J.: *RFC 760: DoD standard Internet Protocol*, Januar 1980.

[Pos81a] Postel, J.: *RFC 791: Internet Protocol*, September 1981. Obsoletes RFC0760 [Pos80]. See also STD0005 [Pos81b]. Status: STANDARD.

[Pos81b] Postel, J.: *STD 5: Internet Protocol: DARPA Internet Program Protocol Specification*, September 1981.

[PR85] Postel, J. und J. K. Reynolds: *RFC 959: File Transfer Protocol*, Oktober 1985.

[PWF+02] Pearlman, L., V. Welch, I. Foster, C. Kesselman und S. Tuecke: *A Community Authorization Service for Group Collaboration*, 2002.

[PWF+03] Pearlman, L., V. Welch, I. Foster, C. Kesselman und S. Tuecke: *The Community Authorization Service: Status and Future.* In: *Proceedings of the International Conference on Computing in High Energy and Nuclear Physics 2003*, 2003.

[RH07] Red Hat, Inc.: *Red Hat Enterprise Linux AS (v. 3) General Advisories.* [Online]. `https://rhn.redhat.com/errata/rhel3as-errata.html`, July 2007.

[SCFY96] Sandhu, R. S., E. J. Coyne, H. L. Feinstein und C. E. Youman: *Role-Based Access Control Models.* IEEE Computer, 29(2):38–47, 1996.

[Sch04] Schneier, B.: *Secrets & Lies: Digital Security in a Networked World*, Seiten 354–361. John Wiley & Sons, Inc., New York, NY, USA, 2004.

[Sim07] Sim, A.: *Storage Resource Management Working Group.* [Online]. `http://sdm.lbl.gov/srm-wg/`, April 2007.

[SNS88] Steiner, J. G., B. G. Neumann und J. I. Schiller: *Kerberos: An Authentication System for Open Network Systems.* In: *Proceedings of the Winter 1988 Usenix Conference*, Seiten 191–201, February 1988.

[SS75] Saltzer, J. H. und M. D. Schroeder: *The Protection of Information in Computer Systems.* In: *Proceedings of the IEEE*, Seiten 1278–1308, 1975.

[SSP+07] Sim, A., A. Shoshani, T. Perelmutov, D. Petravick, E. Corso, L. Magnoni, J. Gu, P. Badino, O. Barring, J.-P. Baud, F. Donno, M. Litmaath, S. De Witt, J. Jensen, M. Haddox-Schatz, B. Hess, A. Kowalski und C. Watson: *The Storage Resource Manager Interface Specification Version 2.2.* [Online]. `http://sdm.lbl.gov/srm-wg/doc/SRM.v2.2.pdf`, April 2007.

[SvBQL04] Snelling, D. F., S. v.d. Berge und V. Qian Li: *Explicit Trust Delegation: Security for Dynamic Grids.* In: *FUJITSU Sci.Tech.Journal*, Band 40, Seiten 282–294, 2004.

[TBKM05] Toninelli, A., J. Bradshaw, L. Kagal und R. Montanari: *Rule-based and Ontology-based Policies: Toward a Hybrid Approach to Con-*

trol Agents in Pervasive Environments . In: *Proceedings of the Semantic Web and Policy Workshop*, November 2005.

[TWE+04] Tuecke, S., V. Welch, D. Engert, L. Pearlman und M. Thompson: *RFC 3820: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile.* [Online]. `http://www.ietf.org/rfc/rfc3820.txt`, June 2004.

[VCF+00] Vollbrecht, J., P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege und D. Spence: *RFC 2904: AAA Authorization Framework.* [Online]. `http:www.ietf.org/rfc/rfc2904.txt`, August 2000.

[VM01] Viega, John und Gary McGraw: *Building Secure Software. How to Avoid Security Problems the Right Way.* Addison-Wesley Longman, Amsterdam, 2001.

[WAM+03] Welch, V., R. Ananthakrishnan, S. Meder, L. Pearlman und F. Siebenlist: *Use of SAML in the Community Authorization Service.* [Online]. `http:http://xml.coverpages.org/WelchSAML20030819.pdf`, August 2003.

[WCHK97] Wahl, M., A. Coulbeck, T. Howes und S. Kille: *RFC 2252: Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions*, Dezember 1997. Status: PROPOSED STANDARD.

[WFK+04] Welch, V., I. Foster, C. Kesselman, O. Mulmo, L. Pearlman, S. Tuecke, J. Gawor, S. Meder und F. Siebenlist: *X.509 Proxy Certificates for Dynamic Delegation.* Proceedings of 3rd Annual PKI R&D Workshop, 2004.

[Wil07] Wilke, O.: *Design and Implementation of a User-Interface for the Creation of User-Based Policies in gLite Environments.* Masterarbeit, Gottfried Wilhelm Leibniz Universität Hannover, 2007.

Tabellarischer Lebenslauf

Stefan Piger

Persönliche Daten

06. Juni 1975	geboren in München als Sohn von Dr. med. Augustin Piger und Dr. med. Ingrid Piger, geborene Wilckens.

Schulbesuch und Ausbildung

1981–1983	Grundschule Nordholz.
1983–1985	Grundschule Otterndorf.
1985–1987	Hauptschule mit Orientierungsstufe Otterndorf.
1987–1994	Gymnasium Otterndorf.
1994–2001	Technische Universität Carolo-Wilhelmina zu Braunschweig, Studium der Elektrotechnik, Schwerpunkt Nachrichtensysteme.
Juli 2001	Abschluss mit der Diplomhauptprüfung.

Wissenschaftlicher Werdegang

seit Juli 2001	Gottfried Wilhelm Leibniz Universität Hannover, Lehrgebiet Rechnernetze, Wissenschaftlicher Mitarbeiter.
2005–2008	Gottfried Wilhelm Leibniz Universität Hannover, Lehrgebiet Rechnernetze, Mitarbeit im BMBF-Projekt *D-Grid-Integrationsprojekt* (DGI) Fachgebiet 3-4 „Authentifizierungs- und Autorisierungsinfrastrukturen“(AAI)
seit Januar 2008	Gottfried Wilhelm Leibniz Universität Hannover, Lehrgebiet Rechnernetze, Mitarbeit im BMBF-Projekt *D-Grid-Integrationsprojekt 2* (DGI-2) Fachgebiet 3-1 „Koordination und Sicherheitsmanagement“