INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Fortgeschrittenenpraktikum

Beurteilung der Sicherheit von Proxy Zertifikaten im Globus Toolkit

Christian Schulz

INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Fortgeschrittenenpraktikum

Beurteilung der Sicherheit von Proxy Zertifikaten im Globus Toolkit

Christian Schulz

Aufgabensteller:	Prof. Dr. D. Kranzlmüller
Betreuer:	Dr. Schiffers Dr. Heller (LRZ) Herr Laitinen (LRZ)
Abgabetermin:	September 2010

Hiermit versichere ich, dass ich die vorliegende Diplomarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 21. September 2010

. . . . . .
(Unterschrift des Kandidaten)

Abstract

Der Bereich Grid Computing gewinnt im Feld der Informatik immer mehr an Bedeutung. Sogenannte 'Grand Challenges' aus den Bereichen Wissenschaft und Ingenieurwesen sind nur mit Hilfe vieler, gekoppelter High Performance Computing Ressourcen zu lösen. Eine Möglichkeit, das dafür nötige Ressource Sharing durchzuführen, sind eben Grids. Um in Grid Umgebungen zu arbeiten, ist eine Middleware nötig. Diese stellt verschiedene Funktionalitäten, wie besispielsweise das Job Management, bereit.
Im Globus Toolkit wird ein Teil der Anforderungen an das Job Management mit Hilfe von Proxy Zertifikaten erfüllt. In den letzten Jahren wurde das dabei verwendete Proxy-Konzept immer wieder kritisiert und als nicht sicher genug für den Einsatz im industriellen Bereich erklärt.
In dieser Arbeit wurden nun anfallende Kritikpunkte gesammelt, in Anforderungen an das Proxy-Konzept des Globus Toolkit Version 4 übersetzt und diese Anforderungen dann überprüft. Dafür wurden auch verschiedene Tests auf Grid Ressourcen des Leibniz Rechenzentrum durchgeführt.

Inhaltsverzeichnis

1 Einleitung

Der Bereich Grid Computing gewinnt im Feld der Informatik immer mehr an Bedeutung. Ein Grund dafür sind die sogenannten 'Grand Challenges'. Hierbei handelt es sich um fundamentale Fragestellungen aus Wissenschaft und Ingenieurwesen, die nur mit Hilfe der Rechenkraft vieler, gekoppelter High Performance Computing Ressourcen gelöst werden können. Komplexe und umfassende Sachverhalte, wie beispielsweise die Bevölkerungsentwicklung, oder das Klima, sollen dabei in Echtzeit verstanden werden.
Ein weiterer Grund für den Aufstieg des Grid Computing ist die Art und Weise, wie Wissenschaft heutzutage betrieben wird. Der Begriff e-Science bezeichnet die Kombination aus Theorie, Experiment und Simulation. Die Analyse von Daten spielt dabei eine entscheidende Rolle und die Simulationen erfordern häufig mehr Rechenleistung, als ein einzelner Großrechner bereitstellen kann.
Der Anstieg an komplexen Fragestellungen, die mit Hilfe von Grids bearbeitet werden sollen, ist besonders deutlich an den Anfragen für Rechenzeit bei der DEISA Extreme Computing Initiative[1] (DECI) zu sehen. Diese Initiative fördert Anwendungen für 'Grand Challenges' aus allen Bereichen der Wissenschaft. Voraussetzung ist die Beschäftigung mit komplexen und innovativen Simulationen, die ohne die DEISA Infrastruktur nicht umgesetzt werden könnten. Im Jahr 2005, dem ersten Jahr der Initiative, wurden dabei Anfragen für 30 Millionen CPU-Stunden gestellt, 2008 bereits für 134 Millionen CPU-Stunden und aktuell, im Jahr 2010, sind es 570 Millionen [1, Folie 11f].
Aufgrund der an sie gestellten Anforderungen weisen Grid Umgebungen besondere Merkmale auf. Sie ermöglichen das kooperative Arbeiten über Organisationsgrenzen hinaus und bestehen daher im Normalfall aus mehr als einer administrativen Domäne. Sie unterliegen also keiner zentralen Kontrolle. Ressource Sharing findet in dynamischen, multiinstitutionalen virtuellen Organisationen statt und soll dabei mit größtmöglicher Sicherheit durchgeführt werden.
Grid Middlewares stellen die benötigten Werkzeuge und Funktionen zur Verfügung, um in solchen Umgebungen zu arbeiten. Ein wichtiger Bereich ist dabei das Job Management. Ressourcen sollen die Möglichkeit haben, als Stellvertreter des Nutzers zu handeln, ohne dass dafür explizite Interaktion nötig wird. Diese dynamische Delegation von Nutzerrechten wird beispielsweise beim Einsatz von Metaschedulern, oder auch beim Abschicken von Jobs, die Filetransfers von anderen Ressourcen benötigen, unerlässlich. Sie wird im Globus Toolkit[2] (GT) mit Hilfe von Proxy Zertifikaten realisiert.
Mit steigender Popularität werden Grids auch für die Nutzung im industriellen Bereich interessanter. Unternehmen auf dem freien Mark legen, aufgrund der Sensitivität ihrer Daten und Projekte, oft noch höhere Sicherheitsstandards an, als das im Bereich der Wissenschaft der Fall ist. In diesem Zusammenhang wurde immer wieder Kritik an der Verwendung von Proxy Zertifikaten im Globus Toolkit laut.
In dieser Arbeit werden die einzelnen Kritikpunkte nun anhand von Globus Toolkit Version

[1] www.deisa.eu/science/deci
[2] www.globus.org

4 validiert, die Sicherheit des Proxy-Konzepts bewertet und gegebenenfalls Vergleiche mit Mechanismen anderer Grid Middlewares angestellt.

2 Globus Toolkit Security

Um eine Bewertung der Sicherheit von Proxy Zertifikaten im Globus Toolkit durchführen zu können, ist zuerst ein Blick auf die Grundlagen des Globus Sicherheitskonzepts nötig.

2.1 PKI und GSI

Das Globus Toolkit implementiert die in Abbildung 2.1 dargestellte Grid Security Infrastructure (GSI) [1]. Dabei wird Secure Sockets Layer[2] (SSL) zur Authentifizierung und zur Message Protection genutzt. Credentials werden durch eine Public-Key-Infrastruktur (PKI) bereitgestellt. Jeder Nutzer, jeder Host, sowie jeder Service benötigt ein X.509 Zertifikat, das von einer vertrauenswürdigen Certificate Authority (CA) signiert ist. Diese Zertifikate werden End Entity Certificates (EECs) genannt.

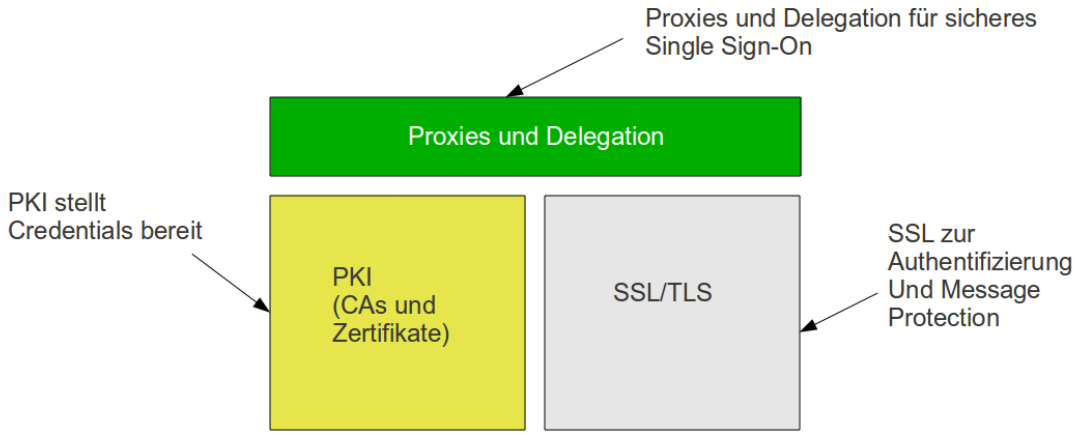

Abbildung 2.1: Grid Security Infrastructure

Jeder Grid-Transaktion geht eine wechselseitige Authentifizierung unter Verwendung von SSL und den EECs, beziehungsweise den in dieser Ausarbeitung genauer untersuchten Proxy Zertifikaten, voraus. Wie in Abbildung 2.2 zu sehen, schickt der eine Kommunikationspartner (X) dem anderen (Y) sein Zertifikat. Handelt es sich um ein EEC, so muss nur die Signatur

[1] www.globus.org/security/overview.html
[2] www.ssl.de/ssl.htm

der CA verifiziert werden. Bei einem Proxy Zertifikat hingegen muss Y erst die komplette Proxy-Kette und im Anschluss die Signatur der CA verifizieren. Wurde dies erfolgreich gemacht, so schickt Y eine zufällige Phrase, den sogenannten 'challenge String', zu X. X wiederum verschlüsselt diese mit seinem privaten Schlüssel und schickt das Ergebnis zurück. Y entschlüsselt daraufhin die Phrase mit dem zuvor im Zertifikat erhaltenen öffentlichen Schlüssel von X und vergleicht sie mit dem Original. Ist der Vergleich positiv, so weiß Y nun, dass es tatsächlich mit X kommuniziert. Für eine wechselseitige Authentifizierung von X und Y muss die komplette Prozedur nun auch noch in die entgegengesetzte Richtung durchgeführt werden.

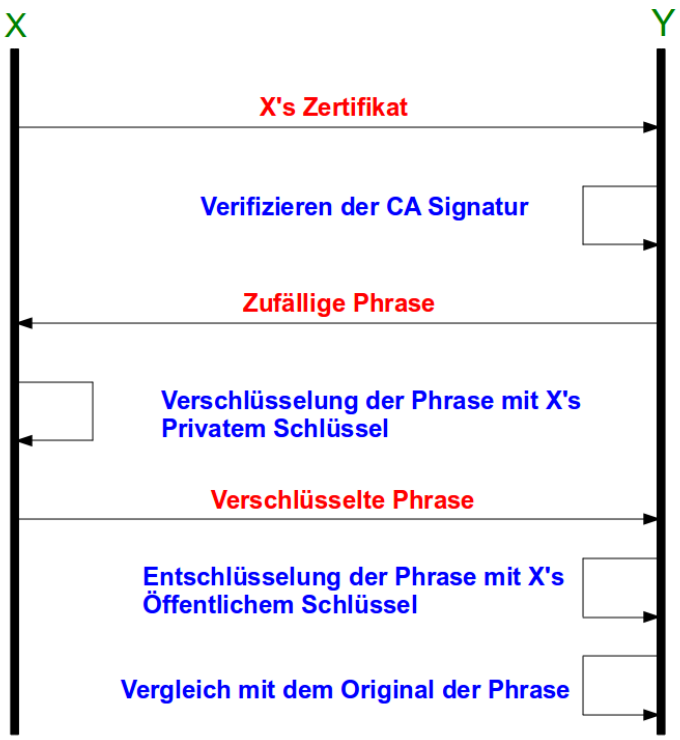

Abbildung 2.2: Wechselseite Authentifizierung

GSI beinhaltet außerdem die Delegation und ermöglicht damit auch einen sicheren Single Sign-On. Das standard SSL Protokoll wird dafür erweitert und mit Hilfe von sogenannten Proxy Credentials, also Proxy Zertifikaten und den dazugehörigen privaten Schlüsseln, können mehrmalige Passwortabfragen vermieden werden.
Diese zentralen Punkte werden im folgenden Abschnitt genauer ausgeführt.

2.2 Proxy Zertifikate

In Kapitel 2.1 wurde das Sicherheitskonzept des Globus Toolkit, und somit auch Proxy Zertifikate als ein Teil davon, vorgestellt. In diesem Kapitel wird nun erläutert, warum Proxies hier verwendet werden und welche Eigenschaften sie besitzen.

2.2.1 Motivation

Einige Anforderungen an eine Grid Middleware legen die Verwendung von Proxy Zertifikaten im Globus Toolkit nahe. So soll beispielsweise ein sicheres Single Sign-On ermöglicht werden, um die Arbeit der Nutzer zu erleichtern und den privaten Schlüssel des EECs zu schützen. Nach einmaliger Eingabe des Passwortes für diesen Schlüssel sollen keine weiteren Passwortabfragen erfolgen.
Eine weitere Forderung ist die dynamische Delegation. Um die Abarbeitung von Workflows möglich zu machen, soll zur Laufzeit entschieden werden können, wer auf welche Ressource zugreifen darf und Ressourcen müssen die Möglichkeit haben ohne explizite Handlung des Nutzers als dessen Stellvertreter aufzutreten. Wird ein GRAM Job an eine Ressource geschickt, so soll diese zum Beispiel im Namen des Nutzers Daten von anderen Servern nachladen können. Die steigende Popularität des Late Binding in Job Management Frameworks erfordert außerdem die Möglichkeit, erst nach dem Abschicken des Jobs die Entscheidung zu treffen, auf welcher Ressource dieser ausgeführt wird. Dies wird nötig, wenn Jobs an einen Metascheduler gesendet werden, der dann, anstelle des Nutzers, entscheidet, an welche Ressource der Job endgültig geschickt wird. Mit Hilfe von Proxy Credentials können diese Anforderungen erfüllt werden.
Außerdem wird, wie bereits erwähnt, durch die Nutzung von Proxies der private Schlüssel des EECs geschützt. Er muss nur einmal zur Erstellung des Proxy Credentials benutzt werden, das ab diesem Zeitpunkt für alle Transaktionen genutzt wird. Das Passwort des privaten Schlüssels wird dabei zu keiner Zeit, weder verschlüsselt, noch unverschlüsselt, über das Netzwerk geschickt. Von verschiedenen Komponenten des Globus Toolkit werden von diesem lokal erzeugten Proxy zusätzliche Proxy Zertifikate abgeleitet und somit die Rechte des Nutzers weiter delegiert. Beispielsweise wird beim Einloggen mittels gsissh auf einem anderen Host für die neue Sitzung standardmäßig ein Proxy Credential erstellt, um damit auf der entfernten Maschine arbeiten zu können.

2.2.2 Eigenschaften von Proxy Zertifikaten

Proxy Zertifikate repräsentieren einen Stellvertreter einer Entität, beziehungsweise ihres EECs. Sie ermöglichen eingeschränktes Proxying und Delegation in einem System, das auf einer Public Key Infrastruktur basiert.
Sie stellen eine eigene Identität dar und sind eine Erweiterung zu X.509 Zertifikaten. Der Distinguished Name (DN) eines neuen Proxy Zertifikats wird aus dem DN des Ausstellers abgeleitet. Ein Beispiel eines Proxy Zertifikats ist in Listing 2.1 zu sehen. Hier kann man den Unterschied zwischen dem DN des Issuers und dem des Subjects erkennen. Dem DN des Issuers wird einfach ein neuer Common Name (CN) angehängt. Wird vom aktuellen Proxy Zertifikat ein weiteres abgeleitet, so kommt ein weiterer CN hinzu. Die Zertifikate enthalten eine neue Extension, die sogenannte ProxyCertInfo, die sich in Listing 2.1 hinter der OID 1.3.6.1.5.5.7.1.14 versteckt.
Proxy Zertifikate sind in RFC3820 standardisiert worden und können sowohl von EECs, als auch von anderen Proxy Zertifikaten signiert werden. Im Gegensatz zu herkömmlichen X.509 Zertifikaten sind sie also nicht von einer Certificate Authority signiert und können deshalb auch nicht über Certificate Revocation Lists (CRL) gesperrt werden.
Für jedes neu erstellte Proxy Credential wird ein Schlüsselpaar erzeugt. Der private Schlüssel ist dabei nicht durch ein Passwort, sondern lediglich durch die Zugriffsrechte des jeweiligen

Dateisystems gesichert. Die Gültigkeitsdauer der Zertifikate wird deshalb, im Vergleich zu EECs, stark verkürzt. Standardmäßig beträgt sie zwölf Stunden. Ein Proxy Credential besteht aus dem Proxy Zertifikat, dem zugehörigen privaten Schlüssel und der gesamten Zertifikatskette, mit Ausnahme der CA-Zertifikate, die direkt bei der CA erhältlich sind. Damit kann das neue Proxy Zertifikat bis hin zur Certificate Authority validiert werden.
Es gibt drei verschiedene Möglichkeiten der Rechtedelegation mittels Proxies. Der Stellvertreter kann entweder alle, keine, oder eingeschränkte Rechte vom Aussteller erben. Die Art der Delegation wird in der oben genannten Erweiterung, ProxyCertInfo, festgelegt. Diese kritische Erweiterung kennzeichnet das Zertifikat als Proxy Zertifikat. Es gibt ein Feld, um die Länge der Zertifikat-Kette festzulegen, die von diesem Proxy abgeleitet werden kann (pCPathLenConstraint). In einem weiteren Feld (proxyPolicy) können die Art der Delegation (policyLanguage) und die Einschränkungen der Rechte (policy) festgelegt werden.

Listing 2.1: RFC3820 Proxy Zertifikat

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1535059767 (0x5b7f2737)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=DE, O=GridGermany, OU=SLCS, OU=Leibniz-
   Rechenzentrum, CN=Christian Schulz - lu23xen@lrz.de
        Validity
            Not Before: Jun  9 11:29:51 2010 GMT
            Not After : Jun  9 23:34:51 2010 GMT
        Subject: C=DE, O=GridGermany, OU=SLCS, OU=Leibniz-
   Rechenzentrum, CN=Christian Schulz - lu23xen@lrz.de, CN
   =1535059767
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (512 bit)
                Modulus (512 bit):
                    00:b1:35:50:e1:6c:6c:b9:f9:70:31:2e:26:ce
   :52:
                    9c:b6:7a:53:fc:fc:0a:38:14:20:85:e3:3e
   :16:41:
                    4b:34:54:05:6b:bd:07:46:9e:84:2e:9d:fc
   :71:62:
                    ea:13:da:15:4b:16:d4:79:03:4d:8b:9d:09:2b:2
   a:
                    18:9d:78:b7:6d
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Extended Key Usage:
                TLS Web Client Authentication
```

```
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment, Data...
            1.3.6.1.5.5.7.1.14: critical
                0.0
..+.......
    Signature Algorithm: md5WithRSAEncryption
    ...
```

2.2.3 Verwendung in Globus Toolkit 4

In den verschiedenen Versionen des Globus Toolkit werden drei Ausprägungen von Proxy Zertifikaten verwendet. 'Legacy Proxy Zertifikate' wurden mit der Version 2.0 eingeführt und werden bis zur Version 4.x unterstützt. Sie enthalten keine ProxyCertInfo Erweiterung und nutzen `CN=proxy`, oder `CN=limited proxy`, um die neue Identität von der des Ausstellers abzuleiten. 'Proxy Draft Proxy Zertifikate' wurden in GT 3 eingeführt und sind den 'RFC3820 Proxy Zertifikaten' sehr ähnlich. Sie enthalten die ProxyCertInfo Erweiterung und sind Standard in den 4.0.x Versionen. 'RFC3820 Proxy Zertifikate' sind komplett konform zum RFC3820 der Internet Society. Sie wurden mit der Version 4 des Globus Toolkit eingeführt und sind Standard in GT 4.2.x.
Um die Funktionen des Toolkits zu nutzen, wird zu Beginn jeder 'Session' ein selbst-signierter Proxy auf dem eigenen System erzeugt. Dies geschieht mit dem Befehl `grid-proxy-init`. Der neue Proxy wird für alle weiteren Transaktionen des Nutzers verwendet und ermöglicht somit den Single Sign-On, da ab diesem Zeitpunkt das EEC des Nutzers nicht mehr verwendet wird und somit keine weitere Passworteingabe nötig ist.
Der `grid-proxy-init` Befehl stellt zahlreiche Optionen zur Verfügung, mit deren Hilfe beispielsweise zwischen den Proxyvarianten gewählt werden kann. Die Standardvariante erhält man mit dem Befehl alleine, die zusätzlich unterstützten mittels Optionen (GT 4.0.x: keine Option für Proxy Draft, -rfc für RFC3820, -old für Legacy).
Die Art der Rechtedelegation zum Proxy wird auch über Optionen geregelt. Standardmäßig werden mit `grid-proxy-init` alle Rechte des Nutzers zum Proxy delegiert. Mit der Option `-limited` werden alle Rechte, bis auf die zum Starten neuer Prozesse, übertragen. Das Abschicken neuer GRAM Jobs ist damit also nicht möglich. Ein 'Independent Proxy' wird mittels `-independent` erstellt. Der dadurch erzeugte Stellvertreter erbt keine Rechte, sondern könnte zum Beispiel in Verbindung mit 'attribute assertions' genutzt werden und seine Rechte über die darin enthaltenen Attribute beziehen.
Proxy Zertifikate mit feingranularen Beschränkungen sind, wie bereits erwähnt, mit RFC3820 konformen PCs grundsätzlich möglich. Wegen der Komplexität der Durchsetzung solcher Einschränkungen ist diese Möglichkeit in Globus Toolkit 4 nicht umgesetzt. Auf die Schwierigkeiten, die dabei zu bewältigen wären, werde ich später noch genauer eingehen. Wie Proxy Zertifikate von den einzelnen Komponenten des Globus Toolkit eingesetzt und welche Rechte dabei delegiert werden, wird in Kapitel 3.3 genauer untersucht.

3 Kritik an GT4 Proxy Zertifikaten

Die Besonderheiten von Grid Umgebungen wurden in der Einleitung bereits angesprochen. Ein Teil der daraus resultierenden Anforderungen an eine Grid Middleware wird im Globus Toolkit mit Hilfe von Proxy Zertifikaten erfüllt. Diese ermöglichen einen sicheren Single Sign-On und die dynamische Delegation von Nutzerrechten.
Es wurde allerdings in den letzten Jahren immer wieder Kritik am verwendeten Proxy-Konzept laut. Es handelt sich dabei um Sicherheitsbedenken, die vor allem im Zusammenhang mit der Verwendung im industriellen Bereich stehen.

3.1 Quellen der Kritik

In Vorträgen, wie diesem [2, Folie 30ff], wurde immer wieder darauf hingewiesen, dass die Verwendung von Proxy Zertifikaten nicht den Sicherheitsstandards entspricht, die von industriellen Nutzern verlangt werden. Was genau am Proxy-Konzept des Globus Toolkit, und der damit verbundenen Delegation von Rechten, die Bedenken auslöst, wurde zumeist nicht genauer erläutert.
Um die Hintergründe der Kritik zu erfahren und die einzelnen Punkte betrachten und validieren zu können, wurde das High Performance Computing Center Stuttgart (HLRS) kontaktiert, da von dieser Seite häufig die Unvereinbarkeit des Sicherheitskonzeptes des Globus Toolkit mit den Sicherheitsstandards aus der Industrie angemerkt wurde.
Dr.-Ing. Stefan Wesner, Deputy Director und Head of Applications and Visualization am HLRS und Prof. Dr.-Ing. Sabine Roller, ehemalige Mitarbeiterin des HLRS und mittlerweile für die German Research School for Simulation Sciences GmbH und das Forschungszentrum Jülich (FZJ) tätig, legten die, ihrer Meinung nach, wichtigsten Kritikpunkte dar. Diese Punkte wurden dann in Sicherheitsanforderungen an Globus Toolkit 4, und speziell an die Verwendung von Proxy Zertifikaten, übersetzt und auf Erfüllung validiert.

3.2 Anforderungen der Kritiker an Proxy Zertifikate

Um die gefundenen Anforderungen zu strukturieren, wurden sie in verschiedene Kategorien eingeteilt. Unterschieden wird dabei zwischen eher technischen Fragen, Anforderungen mit juristischem Hintergrund und Themen im Bereich Nachvollziehbarkeit.

3.2.1 Technik

Es müssen Möglichkeiten zur Einschränkung von Proxies vorhanden sein

Diese Anforderung betrifft die Möglichkeit, die Rechte eines Proxy Zertifikats zu beschränken. Ein Stellvertreter sollte nicht die vollen Rechte des Nutzers erhalten, sondern nur die, die er für seine Aufgabe benötigt. Die Einschränkung der Vollmachten auf einen im Voraus festge-

legten Workflow, der vom Proxy nicht eigenmächtig verändert werden kann, soll ermöglicht werden.

Möglichkeiten zur Verlängerung der Gültigkeitsdauer von Proxy Zertifikaten müssen gegeben sein

Da Proxy Credentials jeweils einen eigenen privaten Schlüssel haben, der nur durch Dateisystemberechtigungen geschützt ist, ist seine Lebensdauer so kurz wie möglich zu halten. Für den Fall, dass sich ein Job länger in der Warteschlange befindet, oder für seine Ausführung mehr Zeit benötigt wird, als ursprünglich angenommen, muss eine Möglichkeit gegeben sein, die Gültigkeitsdauer des Zertifikats nachträglich zu verlängern. Andernfalls entstehen unnötige Kosten, die entweder vom Nutzer, oder vom Betreiber der Ressource getragen werden müssten.

Möglichkeiten zum Zerstören/Stoppen von ausgestellten Proxy Zertifikaten müssen gegeben sein

Hier gibt es zwei Probleme. Die Frage ist zum einen, was mit Proxy Zertifikaten passiert, wenn das EEC, dass sie signiert hat, widerrufen wird und auf die Revocation List (CRL) kommt. Die Aufträge mit diesen Proxies sind ja trotzdem nach wie vor unterwegs. Außerdem sollte es die Möglichkeit geben, ein kompromittiertes Proxy Zertifikat zu widerrufen und somit ungültig zu machen. Über Revocation Lists ist das nicht möglich, da die Zertifikate nicht von einer CA signiert sind.

Nach Beendigung eines Jobs müssen Eingabedaten und Ergebnisse auf allen an der Transaktion beteiligten Stationen gelöscht sein

Hier geht es einerseits um die technischen Möglichkeiten, übertragene Eingabedaten und erhaltene Ergebnisse nach Beendigung eines Workflows (und der anschließenden Rückübertragung) auf den beteiligten Ressourcen löschen zu können. Andererseits ergibt sich hier die Frage, welche Stationen bei der Abarbeitung des Workflows überhaupt beteiligt waren und wo überall Daten angefallen sind. Diese Fragestellung fällt dabei mehr in den Bereich Transparenz und Nachvollziehbarkeit.

3.2.2 Nachvollziehbarkeit

Datenbewegungen müssen nachvollziehbar sein und es muss klar sein, wer Rückschlüsse aus abgeschickten Jobs auf den Stand der Entwicklung ziehen kann

Für diese Forderungen spielen Nachvollziehbarkeit und Transparenz eine entscheidende Rolle. Es geht darum, wer tatsächlich an Transaktionen beteiligt ist, wohin Daten fließen und wer unter Umständen die Möglichkeit hat, aus Art und Anzahl von abgeschickten Jobs Rückschlüsse auf den Stand der Entwicklung von Projekten zu ziehen.

3.2.3 Juristische Fragestellungen

Es muss geklärt werden können, ob Aktionen im Sinne des eigentlichen Nutzers sind

Hier geht es darum, ob Aktionen, die mit Hilfe von Proxy Zertifikaten ausgeführt werden, auch tatsächlich vom ursprünglichen Nutzer gewünscht sind, oder ob sich Proxies

selbstständig gemacht haben. Von den Kritikern wird in diesem Zusammenhang auf die bessere Lösung des Problems durch Unicore hingewiesen, auf die in der Validierung auch kurz eingegangen wird.

Es muss klar sein, wer für Aktionen bezahlt, die von Proxy Zertifikaten durchgeführt werden und wie sich die Proxies in bestimmten Situationen verhalten

Dabei handelt es sich um eine ähnliche Problematik, wie beim vorhergehenden Punkt. Es geht hier um die Sicht des Ressourcen Betreibers, der wissen will, wer für durchgeführte Transaktionen verantwortlich ist und somit auch dafür bezahlen muss.

3.3 Validierung der Anforderungen

3.3.1 Technik

Es müssen Möglichkeiten zur Einschränkung von Proxies vorhanden sein

Es wird bemängelt, dass der Besitzer eines Proxy Zertifikats als der eigentliche Nutzer auftreten kann, ohne dass seine Rechte eingeschränkt sind, beziehungsweise im Voraus eingeschränkt werden können.
In Globus Toolkit 4 gibt es aber sehr wohl einige Möglichkeiten, um Einschränkungen zu verwirklichen.
Man kann die Länge der Proxy-Kette beschränken, die von einem Proxy Zertifikat abgeleitet werden kann. Dies wird durch `grid-proxy-init -path-length <l>` erreicht. Für `l = 0` erhält man einen Proxy, der Proxy Zertifikate signieren kann. Von diesen Zertifikaten können dann aber keine weiteren Zertifikate abgeleitet werden. Erhöht man l, so sind dementsprechend längere Ketten möglich.
Getestet wurde der Mechanismus mit Hilfe von Gsissh-Logins. Dabei wurde das lokale Proxy Zertifikat mit verschiedenen Werten für die Pfadlängen-Option erstellt und dann Gsissh-Hops durchgeführt. Bei einer Pfadlänge von null ist, wie in Abbildung 3.1 zu sehen, nur der Sprung zum nächsten Host, beispielsweise vom Laptop zu lxgt2.lrz-muenchen.de, möglich. Versucht man sich von dort aus über Gsissh auf einen anderen Host, wie mac.lrz-muenchen.de, zu verbinden, so erhält man die Meldung, dass die Pfadlänge überschritten wurde. Die Verbindung ist somit nicht möglich. Bei der Erhöhung des Parameters l um eins ist jeweils ein weiterer Sprung bis zur Fehlermeldung möglich.
Verwendet man die Option nicht, so ist die Pfadlänge nicht eingeschränkt und es wären theoretisch beliebig viele Hops möglich.
Eine weitere Möglichkeit der Einschränkung der Rechte stellt die Verwendung von limited Proxy Zertifikaten dar. Wie erwähnt wird dem Proxy dadurch das Recht genommen, neue Prozesse zu starten. Das Abschicken von GRAM Jobs ist mit einem derartigen Zertifikat nicht möglich.
Um das zu testen, sollte, unter Verwendung eines limited Proxy, ein einfacher Job abgeschickt werden, der lediglich das Datum ausgeben sollte. Der Job wurde von der Gegenseite mit folgender Fehlermeldung abgelehnt: `Fault string: Limited proxy is not accepted`. Das Abschicken neuer Jobs ist mit einem limited Proxy also tatsächlich nicht möglich.
Gsissh kann ebenfalls nur mit einem full Proxy und nicht mit einem limited Proxy genutzt werden. Dabei ist es unerheblich, ob beim Einloggen ein Proxy Zertifikat auf der Remote Maschine erzeugt wird, oder nicht. Die Rechtedelegation bei der Verwendung von Gsissh

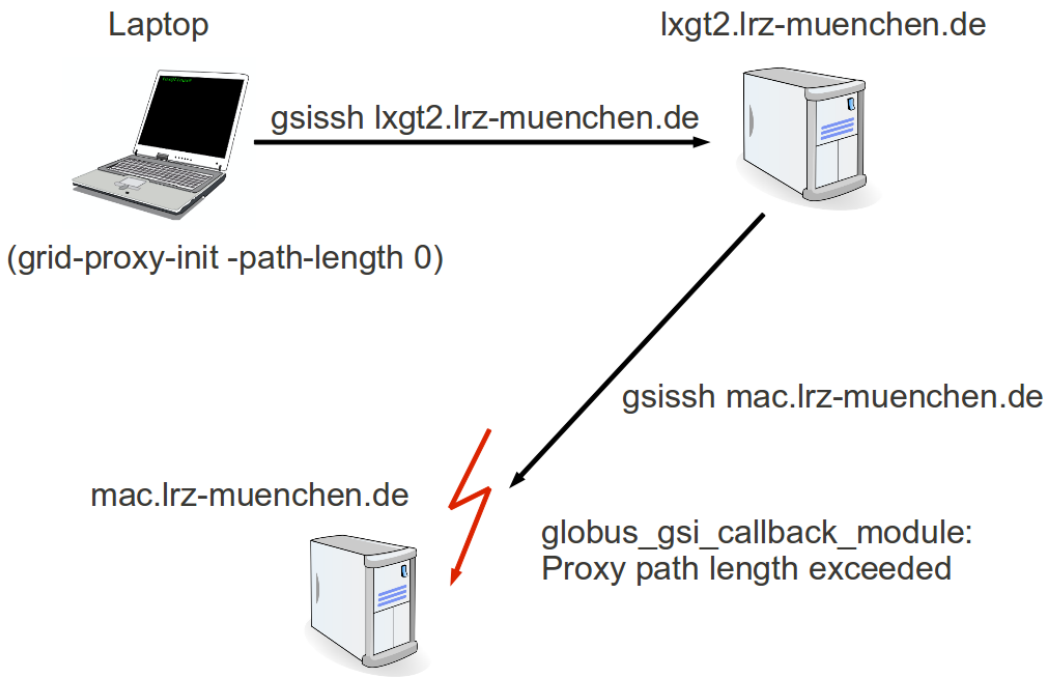

Abbildung 3.1: Gsissh-Hop-Test

wird in diesem Kapitel noch genauer erklärt.
Auch Datentransfers mit GridFTP unter Verwendung von limited Proxies wurden getestet. Diese sind problemlos möglich.
Um die Verwendung der Proxy Zertifikate im Globus Toolkit genauer zu beleuchten, wurde noch betrachtet, in welcher Form einzelne Komponenten Rechte delegieren. Es ging dabei um die Frage, welche Rechte standardmäßig übertragen werden und ob man als Nutzer darauf Einfluss nehmen kann.
Beim Abschicken eines GRAM Jobs kann für den neuen Prozess ein Proxy Zertifikat erstellt werden, das dieser verwenden kann, um weitere Aktionen im Namen des Nutzers auszuführen. Es gibt dabei mehrere Möglichkeiten. Entweder wird einfach ein Proxy vom Zertifikat des Nutzers abgeleitet, oder man nutzt den Delegation Service von GT4 und schickt die Endpoint Reference eines vorher dort gespeicherten Credentials mit. Bei der ersten Variante wird vom GRAM-Client (globusrun-ws) standardmäßig ein limited Proxy vom lokalen Proxy Zertifikat des Nutzers abgeleitet. Verwendet man den Delegation Service, so hängt die Art des Proxys, den der neue Prozess erhält, davon ab, welches Credential vorher beim Service gespeichert wurde. Es ist also auch möglich, dem neuen Prozess einen uneingeschränkten Proxy mitzugeben. Die genaue Funktionsweise des Delegation Service wird im nächsten Abschnitt beschrieben.
Beim Login mit Gsissh wird mit Standardeinstellungen ein Proxy mit vollen Rechten auf dem Remote-Host erzeugt. Der Nutzer meldet sich selbst an einer anderen Maschine an

und will dort im Normalfall auch alle seine Rechte zur Verfügung haben. Es ist aber auch möglich beim Login keinen Proxy erzeugen zu lassen. Dies wird durch Hinzufügen der Zeile `GssapiDelegateCredentials no` in der Datei `~/.ssh/config` erreicht. Nach dieser Änderung führt ein `grid-proxy-info` nach erfolgreicher Anmeldung auf dem Remote Host zu einer Fehlermeldung. Es wird also kein Credential delegiert.
Bei der Nutzung von GridFTP für Third-Party Transfers delegiert der Client Credentials zu den beteiligten GridFTP Servern, damit sich diese auf dem Datenkanal wechselseitig authentifizieren können. Allerdings werden die Credentials im Speicher der GridFTP Server-Instanzen gehalten und nie auf Platte geschrieben. Bei Beendigung des Transfers und damit auch der Server-Instanzen, werden die Credentials also automatisch zerstört.
Die genannten Möglichkeiten lassen zwar Einschränkungen der delegierten Rechte zu, die geforderten Einschränkungen auf bestimmte vordefinierte Workflows sind damit allerdings nicht möglich.
Wie in Kapitel 2 bereits erwähnt, ist im RFC3820 der Proxy Zertifikate die Möglichkeit, Rechte feingranular über Policies zu beschränken, durchaus vorgesehen. In GT wird sie aber nicht umgesetzt. Bei Beschränkungen dieser Art ergeben sich mehrere Probleme. Die Policies müssen derart formuliert sein, dass legitimierte und erwünschte Aufgaben erfolgreich durchgeführt werden können. Außerdem müssen die Ressourcen, beziehungsweise die Enforcement Points, die Regeln interpretieren und ihre semantische Bedeutung durchsetzen können. Es ergibt sich also ein semantisches Problem. Wählt man eine abstrakte Beschreibung der Einschränkungen, so kommt dies zwar dem Nutzer entgegen, ist aber für Ressourcen, beziehungsweise Enforcement Points schwer zu verstehen und zu interpretieren. Ohne globale Kenntnis des kompletten Systems sind solche Regeln nicht auswertbar. Wählt man hingegen eine strenge und genauen Beschreibung, die gut in Regeln umgesetzt werden kann, so müssen viele Informationen vor dem Abschicken des Jobs bekannt sein. Dies ist häufig nicht möglich und ist gegenläufig zur steigenden Popularität des late Binding in Job Management Frameworks. Die richtige Balance zwischen beiden Ansätzen konnte bisher nicht gefunden werden und somit ist das Problem bisher nicht zufriedenstellend gelöst. Die Einschränkung der Rechte eines Proxy auf einen bestimmten Workflow ist also momentan nicht möglich. Die syntaktische Einbindung solcher Regeln hingegen wäre über die in RFC3820 eingeführte ProxyCertInfo Extension möglich.
Unicore 6 nutzt die sogenannte Explicit Trust Delegation (ETD) zur Weitergabe von Rechten. Die Delegation findet hier mit Hilfe von SAML Assertions statt, die im SOAP Header übertragen werden. Diese SAML Trust Delegation Assertions enthalten unter anderem einen Issuer, ein Subject und einen Custodian. Der Issuer gibt die Rechte an das Subject weiter, der Custodian ist bei einer einfachen Rechtedelegation gleich dem Issuer, bei der Verkettung von ETD Assertions der ursprüngliche Issuer, der die erste Assertion ausgestellt hat. Diese Art der Delegation hat, im Vergleich zu den in GT verwendeten Proxy Zertifikaten, den Vorteil, dass die gesamte Delegationskette eingesehen werden kann. Jede beteiligte Zwischenstation ist explizit erkennbar. Bei Proxies ist das, wegen der Ableitung des neuen DN vom DN des ursprünglichen Nutzers, nicht möglich. Die Einschränkung der Rechte auf einen bestimmten Workflow ist allerdings auch beim Mechanismus von Unicore 6 nicht möglich. Es werden alle Rechte des Issuers delegiert. Syntaktisch wären derartige Einschränkungen hier über XACML policys in den SAML Assertion durchaus möglich, sind aber aktuell nicht umgesetzt.

Möglichkeiten zur Verlängerung der Gültigkeitsdauer von Proxy Zertifikaten müssen gegeben sein

Ein Proxy Credential enthält sensitive Daten. Der private Schlüssel ist nur über Dateisystemberechtigungen und nicht zusätzlich durch ein Passwort geschützt. Dies verlangt besondere Sicherheitsvorkehrungen und die Gültigkeitsdauer eines Proxy Zertifikats ist deshalb so kurz wie möglich zu halten. Standard sind zehn bis zwölf Stunden.
Nun kann das aber zu Problemen bei der Ausführung von Jobs mit Hilfe von Proxy Zertifikaten führen, wenn diese in Warteschlangen hängen, oder sich ihre Ausführungszeit unerwartet verlängert.
In Globus Toolkit 4 wurde der sogenannte Delegation Service eingeführt. Er ist über einen Web Service realisiert und bietet die Möglichkeit Credentials zu einem Hosting Environment zu delegieren, die dort dann von allen autorisierten Diensten im selben Container genutzt werden können. Der Nutzer kontaktiert über den Delegation Client (globus-credential-delegate) einen Delegation Factory Service. Es wird ein Credential erzeugt und in einer WS-Resource gespeichert. Die End Point Reference (EPR) dieser Resource wird dem Client zur weiteren Nutzung in einer Datei ausgegeben. Sie kann dann zum Abschicken von GRAM Jobs, oder auch zum Erneuern des gespeicherten Credentials genutzt werden.
Die Erneuerung geschieht über den Delegation Refresh Client (globus-credential-refresh). Dabei wird die zuvor erhaltene EPR des gespeicherten Credentials angegeben. Es wird ein neues Credential erzeugt und das vorher vom Delegation Service erstellte damit überschrieben. Der Service informiert anschließend alle Services, die sich für dieses spezielle Credential registriert haben, über die Änderung und diese holen sich ein neues Zertifikat mit verlängerter Laufzeit. Mit Hilfe des Befehls `wsrf-destroy -e <EPR-File>` kann das gespeicherte Credential zerstört werden.
Durch den Delegation Service erzeugte Credentials können also durch verschiedenen Dienste innerhalb eines Containers genutzt werden und bieten die Möglichkeit des Proxy Renewal. Die Gültigkeitsdauer eines Credentials, das beispielsweise beim Abschicken eines GRAM Jobs genutzt wurde, kann somit nachträglich verlängert werden.
Um die Funktionalität zu testen, wurden mit dem Delegation Service Credentials mit unterschiedlicher Lebensdauer erzeugt und diese beim Abschicken eines GRAM Jobs verwendet. Der Inhalt der dabei verwendeten RSL-Datei ist in Listing 3.1 zu sehen. Die ersten beiden Zeilen legen fest, dass der Prozess zwei Minuten schlafen soll, im Anschluss wird mit Hilfe des `fileStageOut` Tags festgelegt, dass die Datei `test` auf einen anderen Server übertragen wird. Das zuvor erzeugte Credential wird dem Job als Staging- und Transfer-Credential mitgegeben.
Bei der Verwendung eines Credentials mit Standardlebensdauer (zwölf Stunden) wurde der Job ohne Schwierigkeiten ausgeführt. Bei der Beschränkung der Lebensdauer mit `globus-credential-delegate -l 60` auf eine Minute, wurde die Datei nicht übertragen und der Job mit der Meldung `A stagingCredentialEndpoint element was not specified, but is needed for staging` abgebrochen. Zum Zeitpunkt der Übertragung war das Credential nicht mehr gültig.
In einem dritten Test wurde wieder ein Credential mit Lebensdauer eine Minute erzeugt, der Job abgeschickt und im Anschluss mit `globus-credential-refresh -l 600 <EPR>` ein Renewal des Credentials durchgeführt. Die Gültigkeitsdauer wurde damit auf zehn Minuten verlängert. Die Abarbeitung des Jobs verlief problemlos und die Datei wurde übertragen, obwohl das ursprüngliche Zertifikat nur eine Gültigkeit von einer Minute hatte und somit,

durch die zwei-minütige Verzögerung zu Beginn der Abarbeitung, nutzlos gewesen wäre.
Mit Hilfe des Delegation Service des Globus Toolkit ist also durchaus eine nachträgliche Verlängerung der Lebensdauer von verwendeten Credentials möglich.

Listing 3.1: RSL für GRAM Job

```
<job>
 <executable>/bin/sleep</executable>
 <argument>2m</argument>
 <fileStageOut>
   <transfer>
     <sourceURL>file:///${GLOBUS_USER_HOME}/test</sourceURL>
     <destinationURL>gsiftp://mac.lrz-muenchen.de:2811/tmp/
  test</destinationURL>
   </transfer>
 </fileStageOut>
</job>
```

Das Problem an dieser Art des Renewal ist, dass sie auf einen bestimmten Container Beschränkt ist. Werden Jobs an verschiedene Sites geschickt, so muss für jede Site der dortige Delegation Service verwendet und natürlich auch der Renewal mehrmals durchgeführt werden. Praktischer wäre ein Mechanismus, der bei drohender Überschreitung der Gültigkeitsdauer die Erneuerung der Credentials automatisch, und am Besten unabhängig von der Site, an der der Job abgearbeitet wird, durchführt.
Das Workload Management System (WMS) der Grid Middelware gLite[1], das seinen Nutzern erlaubt, Jobs abzuschicken, bietet eine derartige Möglichkeit. Es nutzt dafür das Online Credential Repository MyProxy[2]. Auf einem MyProxy Server können langlebige Credentials sicher gespeichert werden. Nach vorhergehender Authentifizierung können von diesem Credential abgeleitete X.509 Proxy Zertifikate mit kurzer Lebensdauer über das Netz bezogen werden. MyProxy unterstützt dabei verschiedene Arten der Authentifizierung, wie beispielsweise Passwörter, oder Zertifikate. Der private Schlüssel des langlebigen Zertifikats verlässt zu keinem Zeitpunkt den Server.
Der Renewal Service von WMS ist automatisch aktiviert und nutzt standardmäßig den MyProxy Server, der in der dafür vorgesehenen Umgebungsvariable angegeben ist. Soll ein anderer Server genutzt werden, so kann dieser in der Jobbeschreibung explizit angegeben werden. Die Gültigkeitsdauer des Proxies für den abgeschickten Job wird regelmäßig überprüft und bei Bedarf kontaktiert WMS den MyProxy Server. Es wird ein neues Proxy Credential erzeugt und das alte damit ersetzt. Der Job hat also weiterhin ein gültiges Zertifikat und kann abgearbeitet werden. Voraussetzung ist lediglich ein gültiges Credential auf dem MyProxy Server.
Der Mechanismus ist nicht auf eine Site begrenzt, da der MyProxy Server, falls eine Berechtigung vorliegt, von überall genutzt werden kann. Außerdem erfolgt der Renewal automatisch. In GT4 ist solch eine Möglichkeit nicht gegeben. MyProxy ist bereits ins Globus Toolkit integriert und bietet eine Vielzahl von Möglichkeiten in Zusammenhang mit Proxy Zertifikaten. Es wird außerdem ständig weiterentwickelt. Die Nutzung als Teil eines automatisierten Renewal Services in GT wäre daher eine sehr naheliegende Option.

[1] http://glite.web.cern.ch/
[2] http://grid.ncsa.illinois.edu/myproxy/

Auch in Globus Toolkit 5 ist ein automatisierter Renewal mittels MyProxy nicht umgesetzt. Aufgrund der Abkehr vom Konzept der Web Services in dieser Version, entfällt auch die Möglichkeit des Delegation Service. Allerdings ist es dem Nutzer über den Befehl `globusrun -refresh-proxy <URL>` weiterhin möglich, das Credential, das von einem Job-Manager und einem Job genutzt wird, zu erneuern.

Möglichkeiten zum Zerstören/Stoppen von ausgestellten Proxy Zertifikaten müssen gegeben sein

Es wurden zwei Grundszenarien genannt, die zu einer Gefährdung durch Proxy Zertifikate führen können und in denen eine Zerstörung von Nöten wäre. Entweder werden EECs kompromittiert, die noch gültige Proxy Zertifikate signiert haben, oder die PCs selbst werden kompromittiert. Diesen Szenarien gehen üblicherweise Angriffe durch Dritte voraus.
Bevor diese Fälle genauer betrachtet werden, lohnt es sich, einen Blick auf das Verhalten der Globus Komponenten im Normalbetrieb zu werfen. Dabei wurde genauer betrachtet, was mit den ausgestellten Zertifikaten passiert, wenn die Komponenten ihre Aufgabe erledigt haben.
Bei GridFTP findet der Cleanup der PCs automatisch statt. Initiiert ein Nutzer eine Datenübertragung von einem Server zu einem anderen, so erhalten die beiden beteiligten Instanzen der GridFTP Server jeweils ein Proxy Credential, um sich auf dem Datenkanal der Übertragung wechselseitig authentifizieren zu können. Die Credentials werden im Speicher der Instanzen gehalten und nicht auf Platte geschrieben. Bei Beendigung der Übertragung werden auch die beteiligten Instanzen der GridFTP Server beendet und die PCs somit automatisch zerstört.
Beim Login mit Gsissh wird standardmäßig ein full Proxy auf dem Remote Host erzeugt und beim Abmelden automatisch wieder zerstört. Überprüft wurde dieses Verhalten durch eine Anmeldung auf einer entfernten Maschine und der anschließenden Ausführung von `grid-proxy-info`, zur Kontrolle des Speicherorts des delegierten Proxys. Nach dem Abmelden wurde dann die Gsissh-Konfiguration so verändert, dass beim Login kein neues Proxy Credential erzeugt wird. Nach erneuter Anmeldung und der Ausführung von `grid-proxy-info` kam es zu einer Fehlermeldung und am vorher ermittelten Speicherort war das Credential nicht mehr zu finden. Es wurde also offensichtlich beim vorherigen Logout gelöscht.
Bei GRAM ist die Situation etwas komplizierter. Die Delegation von Rechten bei der Nutzung von `globusrun-ws` ist dabei entweder durch das Ableiten eines Proxy Credentials vom Credential des Users, oder durch die Verwendung eines bei einem Delegation Service gespeicherten Credentials möglich. Im ersten Fall wird ein Credential vom PC des Nutzers abgeleitet und im Verzeichnis /$HOME/.globus/ auf der genutzten Ressource gespeichert. Nach der Abarbeitung des Jobs wird ein Cleanup durchgeführt und das delegierte Credential gelöscht.
Es wurden Testjobs an verschiedene Ressourcen des Leibniz Rechenzentrum (LRZ) geschickt und überprüft, ob die Proxies nach der Abarbeitung gelöscht werden. Die Jobs bestanden lediglich aus der Ausgabe der Proxy-Informationen mit dem Befehl `grid-proxy-info`. Damit wurde überprüft, ob die Delegation tatsächlich stattfand. Anschließend wurde durch Gsissh Logins kontrolliert, ob die vorher delegierten Zertifikate tatsächlich weg waren. Der Cleanup wurde bei allen getätigten Tests korrekt durchgeführt und die Credentials waren nach Beendigung der Jobs auf den Ressourcen nicht mehr vorhanden.
Bei Verwendung des Delegation Service zur Rechtedelegation ist im Bezug auf den Cleanup

der Proxies ein anderes Verhalten festzustellen. Hier wurden die selben Jobs abgeschickt, mit dem Unterschied, dass dabei ein, vorher bei den verschiedenen Ressourcen gespeichertes, Credential zum Einsatz kam. Das Job-Credential wurde von diesem abgeleitet und wiederum in /$HOME/.globus/ gespeichert. Allerdings ist das Credential in /$HOME/.globus trotz der Ausgabe von `Cleaning up any delegated credentials...Done.` auch nach Beendigung des Jobs noch vorhanden. Nachdem das Credential in /tmp/ an die Stelle des beim Gsissh-Login delegierten Proxys kopiert wurde, konnte es in vollem Umfang genutz werden. Ein Cleanup der mittels Delegation Service erstellten Credetials wurde also nicht durchgeführt. Dies war auf den verschiedenen getesteten Ressourcen gleichermaßen der Fall.
Zerstört man jedoch mit `wsrf-destroy -e <EPR>` die WS-Resource des Delegation Service, in der vor dem Abschicken der Jobs das Credential gespeichert wurde, so werden auch die für die einzelnen Jobs abgeleiteten Credentials gelöscht. Beim erstellen des Credentials registriert sich GRAM4 beim Delegation Service für einen Callback, falls das Credential dort gelöscht wird. Nach dem Eintreffen des Callbacks führt GRAM den Befehl `globus-gram-local-proxy-tool -delete` aus, um den Proxy im $HOME Verzeichnis des Nutzers zu löschen. Alle getesteten Ressourcen reagierten wie erwartet. Die Credentials wurden gelöscht.
Nach der Behandlung des Standardverhaltens der wichtigsten Komponenten, werden jetzt die genannten Problemfälle betrachtet.
Im ersten Fall geht es dabei um die Frage, was mit Proxy Zertifikaten geschieht, wenn das EEC am Anfang ihrer Proxy-Kette kompromittiert wird. Da die PCs nicht von einer CA signiert sind, können sie nicht direkt über CRLs eingefangen werden. Dies ist im Falle eines kompromittierten EECs aber kein Problem. Sobald dieses auf der Certificate Revocation List steht und diese von den einzelnen Ressourcen regelmäßig aktualisiert wird, sind auch die vom EEC abgeleiteten PCs nicht mehr gültig. Wie in Abbildung 3.2 zu sehen ist, enthält jedes Credential alle Zertifikate der Kette bis hin zum EEC des initialen Nutzers. So enthält das 'Proxy Credential 2' zusätzlich zu 'Bobs Proxy-Zertifikat 2' und dem dazugehörigen privaten Schlüssel noch 'Bobs Proxy-Zertifikat 1' und 'Bobs EEC'. Bei einer Authentifizierung mit 'Bobs Proxy-Zertifikat 2' würde also auch 'Bobs EEC' mit Hilfe des CA Zertifikats validiert. Dabei würde bei aktueller CRL die Ungültigkeit des EECs festgestellt und die Authentifizierung würde fehlschlagen. Der Entscheidende Gefahrenfaktor bei dieser Problemstellung ist also eine nicht aktuelle CRL, nicht aber das Sicherheitskonzept der Proxy Zertifikate im Globus Toolkit.
Komplizierter ist die Lage, wenn nicht die Aussteller-EECs der Proxy Zertifikate kompromittiert werden, sondern die PCs selbst. Dadurch, dass die Proxy Zertifikate nicht einheitlich von Certificate Authorities signiert sind, sondern von EECs, oder anderen Proxies abgeleitet werden, gibt es auch keine zentralen Stellen, um CRLs zu führen. Kompromittierte PCs können also nicht über Credential Revocation Lists, oder ähnliche Mechanismen, gestoppt werden.
Sowohl die Tatsache, dass der private Schlüssel eines Proxy Credentials nicht per Passwort geschützt ist, als auch die fehlende Möglichkeit PCs zu stoppen, macht andere Maßnahmen nötig.
Das Risiko kann über die Laufzeit der Zertifikate minimiert werden. Anders als bei EECs, wo die Laufzeit standardmäßig ungefähr ein Jahr beträgt, ist die Gültigkeitsdauer von Proxy Zertifikaten sehr viel kürzer angesetzt. Im Standardfall ist ein Proxy zwölf Stunden gültig. Da ein kompromittiertes PC keine Auswirkungen auf die Sicherheit des Nutzer-EECs hat, ist die Zeit, in der Gefahr von ihm ausgeht und in der damit Schaden angerichtet werden kann, auf seine Lebensdauer begrenzt. Von einem Proxy abgeleitete PCs können auch maximal die

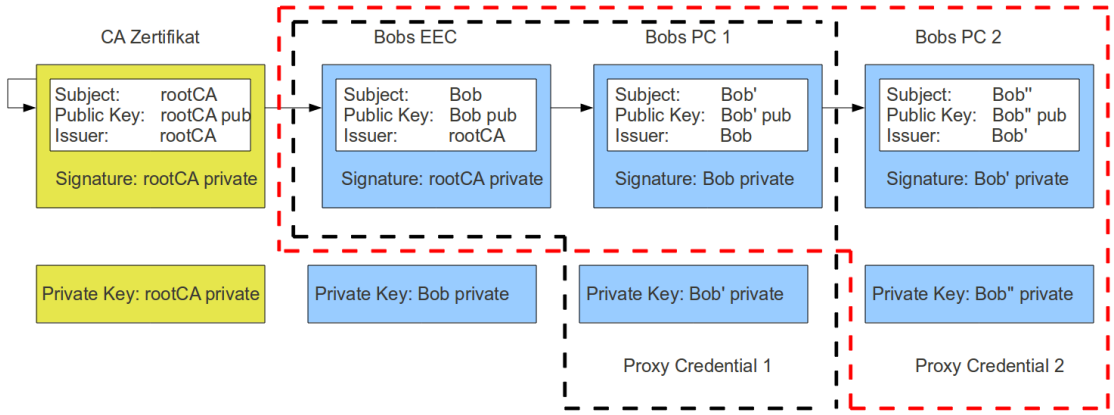

Abbildung 3.2: Aufbau einer Proxy-Kette

Lebensdauer des Aussteller-Zertifikats haben.
Eine längere Gültigkeitsdauer kann allerdings manuell festgelegt werden. Mit dem Befehl `grid-proxy-init -valid <h:m>` kann die Lebensdauer des erstellten Proxys in Stunden und Minuten angegeben werden. Dies ist zwar praktisch für große, lang dauernde Jobs, erhöht aber auch das Risiko, das von den Proxy Zertifikaten ausgeht. Kurzlebige Zertifikate, die bei Bedarf mit Hilfe des Delegation Service verlängert werden, sind die Alternative.
Eine Möglichkeit, um die Sicherheitsrisiken bei der Verwendung von Proxy Zertifikaten im Globus Toolkit zu verringern, wäre die Festlegung einer maximal akzeptierten Gültigkeitsdauer. Würden alle Komponenten auf allen Ressourcen nur noch Proxy Zertifikate bis zu einer bestimmten Gültigkeitsdauer zur Authentifizierung und Autorisierung akzeptieren, so könnte der mögliche Schaden reduziert werden. Diese Forderung ist zwar in Einzelfällen umgesetzt, müsste aber global durchgesetzt werden, um das Konzept der Proxies im Globus Toolkit sicherer zu machen.

Nach Beendigung eines Jobs müssen Eingabedaten und Ergebnisse auf allen, an der Transaktion beteiligten, Stationen gelöscht sein

Grundsätzlich setzt die Ausführung von Jobs auf fremden Ressourcen ein gewisses Vertrauen in diese voraus. Gegebenenfalls können Bedingungen zwischen Nutzern und Dienstanbietern durch Verträge und Service Level Agreements (SLA) festgelegt werden. Hier kann beispielsweise vereinbart werden, was mit den Daten passiert, wenn ein Server vor Beendigung eines Jobs ausfällt. Bei gültigem Zertifikat könnte ein Neustart des Jobs, oder das Zurücksetzen auf einen Sicherungspunkt erfolgen.
Werden Jobs direkt an bestimmte Ressourcen geschickt, so ist klar, wer an ihrer Ausführung beteiligt ist. Wird allerdings ein Metascheduler verwendet, so kommen prinzipiell alle Ressourcen in Frage, zu denen der Nutzer Zugang hat. Der Scheduler entscheidet, welche dieser Ressourcen zum gegebenen Zeitpunkt für die Ausführung am geeignetsten ist. Beim Abschicken des Jobs ist hier also nicht bekannt, wo dieser später ausgeführt wird. Die Transparenz geht also verloren. Dies ist aber kein reines Proxy-Problem, sondern ein Zusammenspiel aus Metascheduling und der fehlenden Möglichkeit, PCs feingranular zu Beschränken.

Manuell ist das Löschen der Eingabedaten und Ergebnisse solcher Aufträge mit den Mitteln des Globus Toolkit durchaus möglich. Im RSL-File eines Jobs können Dateien und Verzeichnisse angegeben werden, die nach der Abarbeitung gelöscht werden sollen. In Listing 3.2 ist die Beschreibung eines Test-Jobs zu sehen. Bei seiner Ausführung soll lediglich das Datum ausgegeben werden. Im Bereich `fileCleanUp` wird festgelegt, dass der results-Ordner im Homeverzeichnis des Globus Nutzers nach der Abarbeitung des Jobs gelöscht werden soll. Der Cleanup-Mechanismus wurde mit Hilfe solcher Test-Jobs auf verschiedenen Ressourcen geprüft und funktionierte überall wie gewünscht. Werden also alle relevanten Daten eines Jobs in bestimmten Ordnern abgelegt, so können diese, nach Beendigung des Jobs und der Übertragung der Ergebnisse zum Nutzer, mit Hilfe des Cleanups gelöscht werden. Anschließend kann bei Bedarf mittels Testjobs kontrolliert werden, ob die Daten tatsächlich gelöscht wurden. Von Cluster-Schedulern werden solche Möglichkeiten häufig auch schon als garantierter Service angeboten. Nicht zuletzt soll damit auch Platzproblemen auf den Clustern vorgebeugt werden.

Listing 3.2: RSL zum Löschen von Ergebnissen

```
<job>
 <executable>/bin/date</executable>
 <fileCleanUp>
   <deletion>
     <file>file:///${GLOBUS_USER_HOME}/results/</file>
   </deletion>
 </fileCleanUp>
</job>
```

3.3.2 Nachvollziehbarkeit

Datenbewegungen müssen nachvollziehbar sein und es muss klar sein, wer Rückschlüsse aus abgeschickten Jobs auf den Stand der Entwicklung ziehen kann

Es muss, wie schon beim vorhergehenden Punkt, unterschieden werden zwischen Aufträgen, die direkt an Ressourcen geschickt werden, und Aufträgen, die über einen Metascheduler gehen. Werden die Jobs direkt über die GRAM-Schnittstelle geschickt, so weiß der Nutzer, welche Ressourcen beteiligt sein werden. Der Metascheduler hingegen sucht aus allen, für den Nutzer möglichen Ressourcen, die geeignetste heraus. Dadurch ist beim Abschicken nicht ersichtlich, wohin die Aufträge und somit auch die Daten geschickt werden. Die beteiligten Stationen sind also nicht bekannt und die Transparenz geht verloren.
Das Proxy-Konzept des Globus Toolkit erfüllt die gestellten Forderungen also erst mal nicht. Es handelt sich hierbei aber nicht um ein reines Proxy-Problem. Vor allem das Late Binding bei der Verwendung von Metaschedulern spielt ebenso eine entscheidende Rolle. Durch das fehlende Wissen um die beteiligten Stationen können die Wege der Daten nicht vorhergesehen werden. Auch eine Einschätzung, wer möglicherweise Rückschlüsse aus den abgeschickten Jobs ziehen könnte, wird erschwert.
Um genau zu wissen, welche Station beteiligt sind, wohin Daten gehen und im Anschluss, wo sie waren, müssten die Rechte, die über Proxy Zertifikate delegiert werden, auf bestimmte Ressourcen beschränkt werden können. Dies führt zur bereits im Kapitel 3.3.1 behandelten

Forderung nach Einschränkung von Proxy Zertifikaten auf bestimmte Workflows. Bei der Formulierung solcher Regeln zur Einschränkung der Rechte wurde bislang kein sinnvolles Gleichgewicht zwischen Anwendbarkeit auf Seiten der Nutzer und Umsetzbarkeit auf Seiten der Enforcement Points gefunden.
Das Proxy-Konzept des Globus Toolkit erfüllt die aufgestellten Forderungen also nicht. Wie bereits in Kapitel 3.3.1 erwähnt, bietet aber auch Unicore 6, zum aktuellen Stand der Entwicklung, keine Möglichkeit zur Delegation von Rechten, beschränkt auf bestimmte Workflows. Bei der Delegation mittels Explicit Trust Delegation werden die Rechte des Nutzers in vollem Umfang weitergegeben. Bei der Verwendung von Metaschedulern sind auch hier sowohl beteiligte Stationen, als auch die Wege der Daten nicht vorhersehbar.
Um den Sicherheitsstandards der Industrie zu genügen, sind in diesem Bereich also noch grundsätzliche Fortschritte nötig.

3.3.3 Juristische Fragestellungen

Es muss geklärt werden können, ob Aktionen im Sinne des eigentlichen Nutzers sind

Es geht hier also darum, ob Aktionen, die im Namen eines Nutzers mit Hilfe von Proxy Zertifikaten durchgeführt werden, von ihm gewünscht sind, oder ob sich Proxies, beziehungsweise deren Inhaber, selbständig gemacht haben. Die Frage ist, wie ein Nutzer den Willen zu bestimmten Aktionen äußert und wie dieser Wille mit Hilfe von Proxy Zertifikaten durchgesetzt wird. Der juristische Begriff hinter dieser Forderung ist die Nichtwiderlegbarkeit. Wenn ein Nutzer einen Job abschickt, darf es ihm nicht möglich sein, dies nachträglich zu leugnen und er muss verantwortlich gemacht werden können.
Beim Start einer Globus-Sitzung wird vom Nutzer ein Proxy angelegt, der den Single Sign-On garantiert und den privaten Schlüssel des Nutzers schützen soll. Dieser Proxy ist nicht eingeschränkt, erhält also die vollen Rechte des Nutzers und ist in dessen Domain von ihm signiert. Von diesem Zertifikat werden dann die Proxies abgeleitet, die für weitere Transaktionen benötigt werden. Es handelt sich dabei im Standardfall um 'limited' Proxys, denen es nicht erlaubt ist, neue Prozesse zu initiieren und die deshalb nicht genutzt werden können, um weitere Jobs abzuschicken. Beispielsweise wird vom globusrun-ws GRAM-Client als Job Credential standardmäßig ein 'limited' Proxy für neue Jobs erstellt. Der einzige unbeschränkte Proxy befindet sich damit in der Domain des Nutzers. Bei Aktionen, die unter Verwendung dieses Zertifikats durchgeführt werden, ist somit klar, dass sie dem Willen des Nutzers entsprechen.
Mit Hilfe des Delegation Service von Globus Toolkit 4 können allerdings auch 'full' Proxys als Job Credentials genutzt werden. Wird bei einem Service ein unbeschränktes Credential gespeichert und dessen EPR beim Abschicken eines Jobs übergeben, so hat dieser Job volle Rechte bei der Ausführung weiterer Transaktionen. In diesem Fall sind auch außerhalb der Domain des Users unbeschränkte Proxys seines EECs unterwegs. Werden mit Hilfe dieser PCs weitere Aufträge abgeschickt, so ist der Wille des Nutzers, dass die damit verbundenen Transaktionen durchgeführt werden sollen, nicht durch eine Signatur in seiner Domain ausgedrückt.
Derartige Aktionen sind bei der Verwendung von Globus Toolkit allerdings nur dann möglich, wenn sich der Nutzer explizit dazu entscheidet, indem er beispielsweise Jobs mit unbeschränkten Job Credentials ausstattet.
Bei Unicore 6 soll die Nichtwiderlegbarkeit nicht wie beim Globus Toolkit durch die Signatur

des zur Authentifizierung genutzten Zertifikats gewährleistet werden, sondern dadurch, dass Jobs und andere wichtige Aktionen eine eigene Signatur erhalten. Und zwar vom sogenannten Consignor. Dieser ist entweder der Nutzer selbst, falls er tatsächlich den Job abschickt, oder eben die Entität, die in seinem Auftrag handelt und dafür Rechte in Form einer Trust Delegation Assertion von ihm erhalten hat. Aktionen, die in Unicore eine Signatur verlangen, sind beispielsweise das Abschicken von Jobs, oder das Zerstören von WS-Resources.
Ein weiterer Unterschied zwischen Unicore und Globus Toolkit besteht in der Authentifizierung selbst. GT verwendet dazu bekanntlich Proxy Zertifikate. Entweder das Sitzungs-Credential, oder eben davon abgeleitete Proxys. Bei Unicore hingegen findet die Authentifizierung immer mit X.509 EECs statt. Dies führt zu hoher Transparenz. Eine Ressource, die eine Jobanfrage erhält, weiß sofort, mit wem sie es zu tun hat. Entweder wird der Job direkt von einem Nutzer geschickt, dann authentifiziert sich dieser mit seinem EEC, oder der Job kommt von einer anderen Entität im Auftrag des Nutzers. In diesem Fall authentifiziert sich diese Entität mit Hilfe ihres EECs und aus der vorgelegten Kette von Trust Delegation Assertions ist der Weg des Auftrags mit allen Zwischenstationen ersichtlich. Bei der Authentifizierung mittels PCs wird nicht klar, wer eine Anfrage endgültig abschickt und welchen Weg sie gegangen ist. Es ist lediglich ersichtlich, wer der ursprüngliche Sender des Auftrags ist und dass dieser seine Rechte delegiert hat.
Durch den Ansatz von Unicore wird Transparenz erreicht und der Wille des jeweiligen Nutzers direkt an seine Aktionen gebunden. Beim Globus Toolkit hingegen wird der Umweg über die verwendeten Proxy Zertifikate genommen und es kann dadurch zu oben genannten Ausnahmen vom normalen Verhalten kommen. Im Standardbetrieb wird aber durch die Nutzung nur eines einzigen unbeschränkten Proxy Zertifikats die Nichtwiderlegbarkeit auch hier gewährleistet. Das Zertifikat befindet sich in der Domäne des Users und ist von dessen EEC signiert.
Grundsätzlich ist die Nichtwiderlegbarkeit ein juristischer Begriff, der nicht von vornherein für alle möglichen, auftretenden Fälle zu hundert Prozent gewährleistet werden kann. Im Zweifel bleibt in diesen Spezialfällen nur der Gang vor ein Gericht, das dann ein Urteil fällt.

Es muss klar sein, wer für Aktionen bezahlt, die von Proxy Zertifikaten durchgeführt werden und wie sich die Proxys in bestimmten Situationen verhalten

Bei der Frage der Bezahlung geht es im Grunde wieder darum, wer für bestimmte Aktionen verantwortlich gemacht werden kann. Der Betreiber einer Ressource muss wissen, wer für anfallende Rechenzeiten, Datenaufkommen, und ähnliches zu belangen ist.
Hat man die Forderung nach der Nichtwiderlegbarkeit erfüllt, so ist bei ankommenden Aufträgen auch klar, wer für die Kosten, die diese verursachen, aufzukommen hat. Beim Ansatz von Globus Toolkit, der die Grid Security Infrastructure umsetzt, soll das, wie im vorhergehenden Punkt ausgeführt, durch die Einzigartigkeit des unbeschränkten Proxy Credentials auf dem Rechner des Nutzers erreicht werden. Ist dieses Sitzungs-Credential als einziges unbeschränkt, so können von den von ihm abgeleiteten Proxys keine neuen Jobs gestartet werden und für alle Jobs, die der Nutzer abschickt, wird zur Authentifizierung ein Zertifikat verwendet, das in seiner Domäne von ihm signiert wurde. Wird dieser Ansatz für die Erfüllung der Nichtwiderlegbarkeit als ausreichend akzeptiert, so ist auch die Frage nach der Bezahlung von Transaktionen leicht zu beantworten.
Im zweiten Teil der Forderung geht es ebenfalls um eine juristische Fragestellung. Es muss geregelt werden, wie sich Proxys in bestimmten Situationen verhalten, was beispielsweise

passiert, wenn ein Job abbricht, oder gekillt wird. Es gibt die Möglichkeit, abgebrochene Jobs, bei noch gültigem Proxy Credential, automatisch neu zu starten, oder eben nicht. Im letzteren Fall müsste der Nutzer selbst reagieren und den Job erneut abschicken. Auch die Frage, wer in diesen Situationen für bereits verbrauchte Rechenzeit aufkommt, ist zu klären. Es handelt sich hierbei aber weniger um proxyspezifische Fragen, als vielmehr um Probleme, die die Systemebene und das verwendete Batch System betreffen. Vereinbarungen über das Vorgehen und Verhalten in bestimmten Situationen müssen beim industriellen Einsatz verhandelt und in Verträgen, oder Service Level Agreements (SLAs) festgehalten werden. Es geht dabei um Vereinbarungen über Rechenzeiten, Datenaufkommen, Verhalten im Fehlerfall und ähnliche Punkte.

4 Bewertung und Fazit

Die eingehende Überprüfung der aufgestellten Forderungen und damit auch der geübten Kritik an der Verwendung von Proxy Zertifikaten im Globus Toolkit Version 4 hat gezeigt, dass kein eindeutiges Urteil möglich ist. Einige der Forderungen werden klar erfüllt, andere nur zum Teil, und wieder andere werden vom Globus Toolkit nicht erfüllt.
Die Möglichkeiten zur Verlängerung eines Proxy Zertifikats beispielsweise sind durch den Delegation Service von GT4 eindeutig gegeben und funktionieren auch wie gewünscht. Eine komfortablere Lösung wäre hier über ein Zusammenspiel des GRAM-Clients mit MyProxy zu erreichen. Dadurch müssten die Verlängerungen der Credentials nicht Site-weise erledigt werden, sondern könnten global und automatisiert durchgeführt werden.
Die Forderung nach Verlängerungsmöglichkeiten ist allerdings auch beim aktuellen Stand bereits erfüllt.
Auch eine Einschränkung der erstellten Proxy Zertifikate ist durchaus möglich. Es kann sowohl die Länge der Proxy-Kette beschränkt werde, als auch die weitergegebenen Rechte limitiert. Die Limitierung verhindert die Schaffung neuer Prozesse. Feingranulare Einschränkungen, zum Beispiel auf festgelegte Workflows, sind bislang leider nicht möglich. Dies stellt ein großes Manko des vorgestellten Ansatzes dar und hat Einfluss auf die Nachvollziehbarkeit von Aktionen. Allerdings bietet beispielsweise Unicore in der aktuellen Version die Möglichkeit der Beschränkung auf Workflows ebenfalls nicht an. Auch hier werden mit Hilfe der Explicit Trust Delegation die vollen Rechte übertragen.
Bei der Forderung nach einer Möglichkeit zum stoppen von Proxy Zertifikaten ist die Antwort zweigeteilt.
Wurde das End Entity Certificate eines Nutzers kompromittiert, so stellen die von diesem Zertifikat abgeleiteten Proxy Zertifikate im Grunde kein Problem dar. Das EEC kommt auf eine CRL und sofern diese auf den Ressourcen regelmäßig aktualisiert werden, werden die PCs damit auch unschädlich gemacht. Bei ihrer Validierung wird das kompromittierte EEC entdeckt und die Authentifizierung scheitert.
Wird ein Proxy Zertifikat selbst kompromittiert, so ist keine direkte Möglichkeit gegeben, dieses zu stoppen. Sicherheit wird hier über eine kurze Laufzeit der PCs erreicht. Allerdings müsste in diesem Punkt noch nachgebessert werden, um die Nutzung von Proxy Zertifikaten mit langer Gültigkeitsdauer zu verhindern. Dies könnte erreicht werden, indem alle Komponenten auf allen Ressourcen nur Zertifikate bis zu einer maximalen Lebensdauer zur Authentifizierung akzeptieren.
Die geforderte Nachvollziehbarkeit, die sowohl die Datenbewegungen als auch die möglichen Rückschlüsse auf den aktuellen Stand der Entwicklung betrifft, wird durch das Proxy-Konzept, bei der Verwendung von Metaschedulern, nicht erfüllt. Durch das Zusammenspiel aus Late Binding und fehlender Möglichkeit der Beschränkung von Proxy Zertifikaten auf Workflows, können die Wege der Daten nicht genau verfolgt werden und es ist nicht klar, wer beispielsweise die Möglichkeit hat, Anzahl und Art abgeschickter Jobs mitzuloggen. Allerdings ist der Kreis der potenziell beteiligten Stationen jeweils auf diejenigen Ressourcen beschränkt, auf die der Nutzer Zugriffsrechte hat. Auch im Fall dieser Forderung existiert

das Problem, aus Mangel an Beschränkungsmöglichkeiten, bei Unicore 6 in selbem Maße.
Nichtwiderlegbarkeit wird beim Proxy-Konzept des Globus Toolkit über die Einzigartigkeit des unbeschränkten Sitzungs-Credentials auf dem Rechner des Nutzers erreicht. Allerdings kann es hier durch die Weitergabe von vollen Rechten über den Delegation Service zu Ausnahmen kommen. Unicore verfolgt hier ein anderes Konzept. Wichtige Aktionen, wie das Abschicken von Jobs, machen hier eine direkte Signatur nötig.
Hat man die Nichtwiderlegbarkeit erreicht, so ist auch die Frage nach der Bezahlung der Rechnungen, die durch Aktionen entstehen, geklärt. Wenn ein Nutzer seinen Willen, eine Transaktion durchzuführen, nicht mehr abstreiten kann, so ist er auch für die Bezahlung der anstehenden Kosten zuständig. Die Vorgehensweise in bestimmten Ausnahmesituationen muss im Vorfeld mit Verträgen, oder Service Level Agreements festgelegt werden. Beispiele wären hier abgestürzte Server, oder abgebrochen Prozesse.
In Sachen Transparenz weist das Proxy-Konzept Mängel auf. Erhält eine Ressource einen Auftrag von einer Entität, die sich mit einem Proxy Zertifikat authentifiziert, so ist für sie nicht ersichtlich, welchen Weg dieser Auftrag gegangen ist. Das PC gibt weder Auskunft über den aktuellen Auftraggeber, noch können die beteiligten Zwischenstationen über die mitgelieferte Proxy-Kette bestimmt werden. Bekannt ist also nur der Nutzer, von dem der Job ursprünglich stammt. Unicore 6 arbeitet hier transparenter. Die Authentifizierung findet ausschließlich mit X.509 EECs statt. Damit ist der aktuelle Auftraggeber bekannt und im Fall eines Stellvertreterauftrags können, über die einzelnen Trust Delegation Assertions der Kette, auch die beteiligten Zwischenstationen erkannt werden.
Dieser Mangel des Proxy-Konzepts könnte beispielsweise durch ein neues Feld in der Proxy-CertInfo Extension behoben werden, das den Inhaber des Proxy Zertifikats als Inhalt bekommt. Allerdings wäre damit eine Abweichung von aktuellen Standards verbunden.
Die meisten der aufgestellten Forderungen werden also vom Globus Toolkit durchaus erfüllt. Einige, wie die Nachvollziehbarkeit der Wege, werden aufgrund ungelöster Problemstellungen nicht eingehalten. Ähnliche Mängel können hier allerdings auch bei Unicore festgestellt werden.
Um die hohen Sicherheitsstandards für den Einsatz in der Industrie einhalten zu können, müssten vermutlich auf dem Gebiet der Transparenz Änderungen vorgenommen werden.
Damit die genauen Vorstellungen und Standards für eine industrielle Nutzung evaluiert und unter Umständen dann besser umgesetzt werden könnten, wäre ein reger Dialog zwischen Entwicklern und interessierten Firmen sicherlich sehr wichtig.

Abbildungsverzeichnis

Literaturverzeichnis

[1] Lederer, Hermann: *DEISA Extreme Computing*. http://www.deisa.eu/news_press/symposium/barcelona2010/presentations/8_DPS-2010-DECI-HermannLederer.pdf.

[2] Riedel, Morris: *SAML - Create and Exchange Security Information in Grids*. http://www.medigrid.de/downloads/080401_security_ws/SAML_Riedel_080402.pdf.