A Security Architecture for e-Science Grid Computing

Vom Fachbereich Informatik der
Technischen Universität Darmstadt genehmigte

Dissertation

zur Erlangung des Grades
Doktor-Ingenieur (Dr.-Ing.)

von

Dipl.-Inform. Steffen Schreiner

geboren in Rothenburg ob der Tauber.

Referenten:	Prof. Dr. Johannes Buchmann
	Prof. Dr. Bernd Freisleben
Tag der Einreichung:	5. November 2014
Tag der Prüfung:	17. Dezember 2014
Hochschulkennziffer:	D 17

Darmstadt 2015

To Julie, Hanne and Gerhard

In memory of Ida and Karl

List of Publications

[PUB1] Costin Grigoras, Latchezar Betev, Pablo Saiz and Steffen Schreiner. Optimization of Grid Resources Utilization: QoS-aware client to storage connection in AliEn. *Proceedings of Science (PoS)*, 13th International Workshop on Advanced Computing and Analysis Techniques in Physics Research, ACAT2010:32, 2010.

[PUB2] Fabrizio Baiardi, Paolo Balboni, Rui Barros, Daniele Catteddu (ed.), Peter Dickman, Theo Dimitrakos, Marcos Gomez, Amanda Goodger, Dennis Heinson, Giles Hogben, Ben Katsumi, Antonio Lioy, Liam Lynch, Philippe Massonet, Srijith Nair, Milan Petkovic, Jim Reavis, Raj Samani, Steffen Schreiner and David Wright. Security & Resilience in Governmental Clouds. *European Network and Information Security Agency (ENISA)*, 2011.

[PUB3] Jianlin Zhu, Pablo Saiz, Federico Carminati, Latchezar Betev, Daicui Zhou, Patricia Mendez Lorenzo, Alina Gabriela Grigoras, Costin Grigoras, Fabrizio Furano, Steffen Schreiner, Olga Vladimirovna Datskova, Subho Sankar Banerjee and Guoping Zhang. Enhancing the AliEn Web Service Authentication. *Journal of Physics: Conference Series (JPCS)*, 331, p. 062048, 2011.

[PUB4] Steffen Schreiner, Stefano Bagnasco, Subho Sankar Banerjee, Latchezar Betev, Federico Carminati, Olga Vladimirovna Datskova, Fabrizio Furano, Alina Grigoras, Costin Grigoras, Patricia Mendez Lorenzo, Andreas Joachim Peters, Pablo Saiz and Jianlin Zhu. Securing the AliEn File Catalogue - Enforcing authorization with accountable file operations. *Journal of Physics: Conference Series (JPCS)*, 331, p. 062044, 2011.

[PUB5] Steffen Schreiner, Latchezar Betev, Costin Grigoras and Maarten Litmaath. A Mediated Definite Delegation Model allowing for Certified Grid Job Submission. *Computing Research Repository (CoRR)*, 2011. [`abs/1112.2444`].

[PUB6] Jianlin Zhu, Pablo Saiz, Latchezar Betev, Federico Carminati, Costin Grigoras, Steffen Schreiner, Alina Gabriela Grigoras, Fabrizio Furano, Daicui Zhou and Guoping Zhang. Grid Architecture for ALICE Experiment. In *Advanced Communication Technology (ICACT), 14th International Conference on*, pp. 1209–1214. IEEE, 2012.

[PUB7] Manfred Alef, Stefano Bagnasco, Jakob Blomer, Andrea Ceccanti, Alessandro De Salvo, Claudio Grandi, Dave Kelsey, Maarten Litmaath, Steffen Schreiner and Igor Sifliogi. WLCG Grid job and Worker Node security assessment. *Worldwide LHC Computing Grid Collaboration, WLCG Technical Evolution Group on Security*, 2012.

[PUB8] Dave Kelsey, Maarten Litmaath, Steffen Schreiner, Von Welch, Romain Wartel, John White and Christoph Witzig. WLCG Computer Security Risks Analysis. *Worldwide LHC Computing Grid Collaboration, WLCG Technical Evolution Group on Security*, 2012.

[PUB9] Steffen Schreiner, Costin Grigoras, Alina Grigoras, Latchezar Betev and Johannes Buchmann. A security architecture for the ALICE Grid Services. *Proceedings of Science (PoS)*, The International Symposium on Grids and Clouds, ISGC2012:027, 2012.

[PUB10] Steffen Schreiner, Costin Grigoras, Maarten Litmaath, Latchezar Betev and Johannes Buchmann. Mediated definite delegation - Certified Grid jobs in ALICE and beyond. *Journal of Physics: Conference Series (JPCS)*, 396, p. 032096, 2012.

[PUB11] Jin Huang, Pablo Saiz, Latchezar Betev, Federico Carminati, Costin Grigoras, Steffen Schreiner and Jianlin Zhu. Grid Architecture and implementation for ALICE experiment. In *Advanced Communication Technology (ICACT), 16th International Conference on*, pp. 253–261. IEEE, 2014.

[PUB12] Alina Gabriela Grigoras, Costin Grigoras, Miguel Martinez Pedreira, Pablo Saiz and Steffen Schreiner. JAliEn - A new interface between the AliEn jobs and the central services. *Journal of Physics: Conference Series (JPCS)*, 523, p. 012010, 2014.

Acknowledgment

First and foremost, I sincerely thank my supervisor Johannes Buchmann as well as my CERN supervisor Latchezar Betev for their enormous support, guidance, challenge and the countless discussions throughout the last years. Also my sincere gratitude goes to Bernd Freisleben for his support and guidance as second referee.

Furthermore, I would like to thank a lot: the ALICE Offline Group and ALICE LCG Task Force, in particular Subho Sankar Banerjee, Federico Carminati, Olga Vladimirovna Datskova, Alina Gabriela Grigoras, Costin Grigoras, Fabrizio Furano, Arsen Hayrapetyan, Maarten Litmaath, Patricia Mendez Lorenzo, Andreas-Joachim Peters, Pablo Saiz, Matevz Tadel, Jianlin Zhu. The CDC research group in Darmstadt and all members of CASED, in particular Özgür Dagdelen, Evangelos Karatsiolis, Michael Kreutzer, Richard Lindner, Markus Rückert, Michael Schneider, Melanie Volkamer, Alexander Wiesmaier and last but not least Marita Skrobić. The members of the WLCG Technical Evolution Groups [PUB7, PUB8], in particular Dave Kelsey and Romain Wartel. And Daniele Catteddu and Jintai Ding, who gave me the chance to reach out beyond the area of Grid Computing Security.

Special thanks go to: Alina and Costin, again, for countless javatized 'Have a cup' meetings that gave rise to the jAliEn. Andreas, again, for giving me the right pointers not only in C++. Till Böhlen, for setting me up with the idea of applying for a doctoral position at CERN and accompanying me during my time there. Christoph Skupnik for challenging my diction and his companionship. Fabian Held and Herta Schwender for carefully reviewing my spelling. All so far unnamed friends, around Darmstadt, Geneva, Berlin and elsewhere for their invalueable support, counseling and encouragement. And Julie, Hanne and Gerhard for everything.

Darmstadt, November 2014 *Steffen Schreiner*

Abstract

E-Science Grid infrastructures are established on the collaboration of multiple and possibly otherwise independent and globally distributed organizations connected via the Internet. Thereby instantiated e-Science Grids provide the researchers of these globally distributed organizations with unified access to large-scale computing and storage services, including the access to large-scale scientific data as such. It is part of their purpose that e-Science Grids allow collaborating researchers to introduce their own data and program code in the course of their work. Beyond, via submission of Grid jobs any program code can be executed as detached computational operation within the distributed computing infrastructure.

This openness of allowed introduction and usage of data and program code poses a substantial security threat. The delegation of privileges in the course of Grid jobs submissions in combination with the users' allowance to introduce and utilize a priori untrusted program code and data is however a widely identified security challenge [78, 61, 121]. The main contribution of this thesis is to propose a new framework for delegation and an according Grid security architecture in response to this challenge.

Following a discussion of the goals and requirements of e-Science Grids in general, and an overview and comparison of existing and in-use e-Science Grid architectures in particular, this thesis will analyze security aspects applying to such e-Science Grid infrastructures. The thereof derived and defined security objectives concern data integrity and authenticity, system integrity, availability, non-repudiation of Grid job submission and processing as well as confidentiality and data privacy.

Looking at the case of an established e-Science Grid framework, vulnerabilities and security implications of existing distributed e-Science Grids will be examined. Furthermore the widely adopted unrestricted delegation based on X.509 proxy certificates [132] will be assessed, revealing fundamental deficiencies concerning the unverifiable correlation of assignment and delegation of privileges, which facilitates potential misuse of privileges and digital identities.

In order to address these issues, this thesis will introduce *"mediated definite delegation"* as a new framework for delegation. The framework utilizes public-key signatures and affords verifiable integrity and authenticity of Grid data and jobs as well as transparent, dynamic and least-privileged delegation of Grid jobs via one or more brokers to an agent. Its delegation mechanism provides protection against misuse of the delegating user's identity as well as against unnoticed alteration of the requested actions.

Finally, an e-Science Grid security architecture established on this framework will be presented and specified, which is able satisfy the defined security objectives. As a proof of concept, a prototype implementation of this e-Science Grid security architecture will be presented, including a test-based evaluation of its performance.

Zusammenfassung

E-Science-Grid-Infrastrukturen basieren auf der Kollaboration einer Vielzahl von unabhängigen und global verteilten Organisationen, welche durch das Internet digital verbunden sind. Dadurch geschaffene e-Science-Grids ermöglichen den Wissenschaftlern dieser Organisationen einen einheitlichen Zugang zu Hochleistungsrechner und -speichersystemen sowie zu umfangreichen wissenschaftlichen Daten. Die Nutzung von eigens eingebrachten Daten und Programmcodes durch Wissenschaftler im Rahmen ihrer Forschungsarbeit stellt dabei einen zentralen Verwendungszweck eines e-Science-Grids dar. Dabei kann die Ausführung beliebiger Programmcodes in Form von Grid-Jobs in Auftrag gegeben werden, welche als automatisierte Prozesse innerhalb der Grid-Infrastruktur verteilt und ausgeführt werden.

Dieser Offenheitsgrad der Verwendung von eigens eingebrachten Daten und Programmcodes stellt eine substantielle Sicherheitsbedrohung dar. Die Delegation von Benutzerprivilegien im Zuge der Versendung von Grid-Jobs in Kombination mit der erlaubten Einbringung von a priori nicht vertrauenswürdigen Daten und Programmcodes ist jedoch eine bekannte Herausforderung innerhalb der IT-Sicherheit [78, 61, 121]. Der zentrale Beitrag der vorliegenden Arbeit ist die Spezifikation eines neuen Delegationsverfahrens und einer darauf aufbauenden Sicherheitsarchitektur für e-Science-Grids als Lösung für dieses Problem.

Nach einer Diskussion der Ziele und Voraussetzungen von e-Science-Grid-Infrastrukturen im Allgemeinen und einem Vergleich bestehender e-Science-Grid-Architekturen werden in der vorliegenden Arbeit die Sicherheitsaspekte von e-Science-Grid-Infrastrukturen analysiert. Die davon abgeleiteten Sicherheitsziele betreffen Datenintegrität und -authentizität, Systemintegrität, Verfügbarkeit, Verbindlichkeit der Entsendung und Verarbeitung von Grid-Jobs sowie Vertraulichkeit und Datenschutz.

Am Beispiel eines bestehenden und genutzten e-Science-Grids werden Schwachstellen und Sicherheitsauswirkungen in Bezug auf das System und die dazugehörige verteilte Infrastruktur untersucht. Darüber hinaus wird das weithin genutzte Delegationsverfahren der unbeschränkten Delegation basierend auf X.509-Proxy-Zertifikaten [132] analysiert. Dabei werden fundamentale Schwachstellen aufgezeigt, die insbesondere die nicht verifizierbare Korrelation von Aufträgen und delegierten Privilegien betreffen und folglich den Missbrauch von Privilegien und digitalen Identitäten ermöglichen.

Als Lösung wird *"mediated definite delegation"* als ein neues Delegationsverfahren basierend auf digitalen Public-Key-Signaturen präsentiert und spezifiziert. Das Verfahren ermöglicht verifizierbare Integrität und Authentizität von Grid-Daten und -Jobs sowie eine transparente, dynamisch vermittelbare und geringst-möglich privilegierende Delegation von Grid-Jobs über einen oder mehrere Vermittler. Weiterhin ermöglicht es den Schutz der digitalen Identität eines delegierenden Benutzers vor Missbrauch und verhindert eine unbemerkte Veränderung der in Auftrag gegebenen Vorgänge.

Schließlich wird eine Sicherheitsarchitektur für e-Science-Grids präsentiert und spezifiziert, welche auf dem Delegationsverfahren basiert und das Erreichen der definierten Sicherheitsziele ermöglicht. Eine Implementierung der Sicherheitsarchitektur als Prototyp wird vorgestellt, und die Leistungsfähigkeit der Implementierung innerhalb einer testbasierten Evaluation bestätigt.

Contents

"The beginning is the most important part of the work."

Plato

1 Introduction

Science utilizing computationally intensive research methods established on the collaboration of research organizations is generally referred to as *e-Science* [27] and applies to many research areas, such as particle physics, medicine, biology and geology. Grid computing is a technical architecture model applied within such cyberinfrastructures and provides a distributed system layer arranged on top of discrete and localized distributed systems of lower order.

With multiple and possibly otherwise independent and globally distributed organizations connected via the Internet, e-Science Grid infrastructures provide researchers with unified access to large scale computing and storage services, including the access to large-scale scientific data as such. In doing so, collaborating researchers are allowed and invited to introduce their own data and program code in the course of their work, and to submit requests for predefined and later-detached computational tasks, which are executed as Grid jobs within the distributed computing infrastructure.

1.1 Motivation

From a technical perspective, e-Science Grid infrastructures do not only interconnect organizations and researchers, but providers and users beyond the borders of the underlying organizational affiliation and the corresponding levels or domains of control and responsibility. Accordingly, e-Science Grid infrastructures are required to facilitate traceability [91, 116] of operations and their users' activities as well as protection against illicit system usage.

However, the necessary delegation of privileges in the course of submissions of Grid jobs, in combination with the users' allowance to introduce and utilize a priori untrusted program code and data, represents a widely identified security challenge [78, 61, 121]. Moreover, the interconnection of entities and players dissolves the classical differentiation of internal and external attackers in view of potential vulnerabilities or malicious insiders in any of the employed systems and layers. Consequently, an e-Science Grid infrastructure cannot only rely on protection against inbound attacks, but must be protected as well against malicious insiders, being used as a platform for outbound malicious actions or facilitating attacks on any of its sub-systems or third party entities.

This thesis presents a new framework for delegation and its incorporation in a Grid security architecture utilizing public-key signatures, affording transparent, dynamic and least-privileged delegation of Grid jobs. The presented Grid security architecture provides protection of a Grid infrastructure against undetectable malicious behavior as well as protection of its users' digital identities and reputations, without constraining or limiting the infrastructure's discretionary and permissive use.

The particle physics experiments of the *Large Hadron Collider (LHC)* [31] and their corresponding international Grid infrastructure project, the *Worldwide LHC Computing Grid (WLCG)* [35], have been a driving force in the development and establishment of international e-Science Grid infrastructures. The *ALICE* [1] experiment, as one of the four large LHC experiments, is the main reference point of the herein presented work, as an existing and in-use e-Science Grid infrastructure. The detector of the experiment is built and operated by an international research collaboration [2] and its computational analysis is based on the *ALICE Grid Services* [3]. As an international e-Science Grid infrastructure, the ALICE Grid Services provide a unified environment for the physics data analysis of the experiment based on more than 70 computing centers of collaborating institutes located in over 30 countries.

Beyond their relevance in the scope of Grid infrastructures and the Grid computing paradigm, the results presented in this thesis have further significance within the field of cloud computing. There, the processing of user- or client-provided code is likewise a basic functionality, e.g. via Infrastructure or Platform as a Service layers. An example are cloud customers building yet another service level on top of utilized cloud infrastructures: in doing so, these customers act at the same time as service providers, while being presumably liable [10] for the transitive usage of a respective cloud infrastructure by their end users.

1.2 Outline and Summary of Results

This thesis is divided into three parts. A general description of current uses and architectural models of e-Science is presented in Chapters 2 and 3 to set the scene for the contributions of this thesis. In Chapter 2, an e-Science scenario is specified, outlining the typical usage and characteristics as well as its relevant assets. Chapter 3 defines a generic architecture for an e-Science Grid, based on a comparison of existing infrastructures.

The second part addresses security aspects of an e-Science environment. Chapter 4 identifies relevant criteria as security objectives and Chapter 5 and 6 present vulnerabilities and deficiencies in existing e-Science Grid infrastructures and methodologies. Chapter 5 will examine the concrete case of the ALICE experiment, and Chapter 6 will analyze the widely deployed unrestricted delegation based on X.509 proxy certificates.

The third part of this thesis presents a solution with respect to the security objectives and deficiencies identified. Chapter 7 introduces a new framework for delegation, which stipulates digitally signed textual statements for both delegation and attestation of data authorship. The following Chapter 8 then presents an e-Science Grid security architecture able to meet the security objectives defined in Chapter 4, which is established on the introduced delegation framework while also including further mechanisms.

This part of the thesis concludes by the presentation of a prototype implementation and its performance evaluation in Chapter 9. The thesis then concludes in Chapter 10, summarizing its contribution and results and discussing future work.

A summary of the research contributions and results of this thesis is presented as follows:

Chapter 2: E-Science Computing Based on the ALICE [1] experiment's e-Science Grid environment and other e-Science infrastructures and approaches, a generic e-Science scenario is defined. Further on, assets with respect to the scenario are specified, based on a generalization and extension of the existing definition of assets within the WLCG [PUB8]. Both scenario and assets will be utilized in particular as the basis for the definition of security objectives in Chapter 4.

The e-Science scenario is summarized as follows: research organizations collaborate within the organizational framework of a Virtual Organization (VO). This VO combines the distributed organizations and respective facilities and achieves their unification within one collaboration and a corresponding framework, instead of concurrent and discrete operations and research. On the basis of this framework, cross-organizational unified access to distributed data and processing infrastructure is provided to its users. This enables the processing of scientific data via submission of predefined tasks (Grid jobs) subsequently executed within the infrastructure, wherein the utilization of user-provided program code and data is facilitated.

Chapter 3: E-Science Grid Architectures On the basis of a specification of the architecture of the ALICE Grid Services [3] and a subsequent comparison with other instantiated e-Science Grid infrastructures, a generic e-Science Grid architecture is defined. This architecture will be utilized in the subsequent chapters of this thesis as the reference architecture for e-Science Grids.

The e-Science Grid architecture is summarized as follows: within the framework of a VO, a virtual Grid layer is established by logical abstraction, connecting a multitude of sub-systems as services and clients via uncontrolled networks, including the Internet. Collaborating institutes as Sites provide computing and storage infrastructure as Worker Nodes and Storage Elements, which are logically unified within a set of Central Grid Services, most importantly providing Grid File Catalogs and Grid Task Queues. Grid users are provided with access to virtual Grid file systems, established by Grid File Catalogs and Site-provided Storage Elements as physical storage devices. Further, Grid users are provided with access to computational resources via the submission of tasks as predefined textual requests, hereafter referred to as Grid jobs, to Grid Task Queues. These are subsequently distributed to Site-provided Worker Nodes and executed without a user's interference or attention.

Chapter 4: Security Analysis of e-Science Grids In reference to the defined e-Science scenario and Grid architecture, the involved players and their relationships, the security characteristics and potential attack motives and targets are specified, and derived security objectives defined. These security objectives will be used in the subsequent chapters of this thesis as general assessment criteria.

E-Science Grids are based on the interactions of three basic groups of players: first, the Virtual Organization (VO) operators, second, the collaborating Sites as providers, and third, the VO's Grid users. Moreover there are third parties like other VOs or organizations as adjacent or co-users of the shared infrastructure as well as umbrella Grid organizations in which several VOs can be embedded in.

The specified security characteristics are an e-Science Grid's public communication environment, the discretionary controlled and non-exclusive Site-provided resources, the discretionary usage by its users and uncontrolled client environments, and the cross-organizational access within the infrastructure. The consequently defined security objectives for e-Science Grids concern data integrity and authenticity, system integrity, availability, non-repudiation of Grid job submission and processing, and confidentiality and data privacy.

Chapter 5: Vulnerability Analysis of the ALICE Grid Services A vulnerability analysis of the ALICE Grid Services as an established in-use e-Science Grid infrastructure is presented, identifying a series of vulnerabilities and deficiencies.

In essence, these affect the infrastructure's inbound and outbound protection and the authenticity and authorship of entities and interactions within the established Grid layer. The vulnerabilities originate in the infrastructure's reliance on unverified or user-provided information concerning authenticity and authorship. Further on, the Grid job processing and execution is based on delegation by masquerade, which allows unverifiable misuse of privileges and digital identities of job submitters.

Chapter 6: Analysis of Delegation and X.509 Proxy Certificates Following an analysis of delegation in Grid computing and X.509 proxy certificates [132] as well as their propagation within a Grid infrastructure, vulnerabilities and deficiencies identified in the widely adopted delegation approach of unrestricted delegation based on X.509 proxy certificates are specified.

The analysis reveals fundamental deficiencies and vulnerabilities concerning Grid job delegation, which most importantly are the unverifiable correlation of assignment and delegation and the dependence on and required trust in VOs and Sites concerning potential misuse. Consequently, unrestricted delegation by utilization of X.509 proxy certificates is assumed to provide no fundamental improvement in comparison to mechanisms implementing delegation by masquerade, as they afford unidentifiable misuse of privileges and digital identities.

Chapter 7: A Framework for Transparent Dynamic Delegation A new delegation framework named *mediated definite delegation* is presented, facilitating transparent, dynamic and least-privileged delegation of tasks.

The framework is based on textual statements for both task delegations and attestation of data authorship, which are required to be signed using public-key signature mechanisms. In case of task delegations, cascaded signatures are utilized within the different steps of submission and processing and propagation, while a task delegation can be processed and propagated via more than one broker to an agent for execution. The mechanisms afford non-repudiation of authorship of data as well as of task delegations and their processing and propagation. Moreover, task delegations as credentials afford protection against misuse of the delegating user's identity as well as against unnoticed alteration of the requested actions.

The framework and its mechanisms integrate completely into the e-Science Grid architecture defined in Chapter 3 and require no remote callbacks or additional invocations.

Chapter 8: A Security Architecture for e-Science Grids Based on the introduced framework of *mediated definite delegation*, a new security architecture for e-Science Grids is presented and specified.

Involving mechanisms with respect to Grid data layer access, storage-confirmed authenticity of Grid file data and a dynamic storage discovery, the security architecture provides certified Grid files via user-signed *File Certificates*. Furthermore, on the basis of an initial signature of a Grid job request by a submitter, and a subsequent additionally applied signature by an e-Science VO as a broker, certified Grid jobs are achieved. The resulting concept of a *Job Certificate* offers verification of an initial submission as well as its later processing as a Grid job and arranges for the utilization of isolation mechanisms for Grid job execution.

The enforced utilization of File and Job Certificates establishes verifiable authenticity of the represented entities and non-repudiation of their authorship. Hence, meaningful decision-making concerning the traceability of Grid files and jobs can be achieved. Beyond the framework of mediated definite delegation, the security architecture for e-Science Grids thereby facilitates the fulfillment of the security objectives defined in Chapter 4.

Chapter 9: A Grid Middleware Prototype Within the Grid middleware project *jAliEn* [22], the introduced security architecture for e-Science Grids was implemented as a prototype, with a focus on the security architectureâĂŹs concepts of Grid job delegation and Job Certificates.

The implementation provides Grid service communication via serialization and exchange of high-level Java objects, established on mutually authenticated and encrypted connections. The implemented prototype represents a proof of concept of the Grid security architecture, with its practical relevance being confirmed within a presented performance evaluation. Despite the employed security mechanisms and digitally signed Grid job submissions, the performance evaluation reveals notably capacity with respect to service response times as well as to scalability and high loads.

"Science knows no country, because knowledge belongs to humanity, and is the torch which illuminates the world."

Louis Pasteur

2 E-Science Computing

In this chapter the e-Science Grid infrastructure of the ALICE [1] experiment as well as other Grid-based e-Science examples and scenarios will be analyzed concerning their utilization as distributed research cyberinfrastructures.

The analysis aims at the understanding of the purpose, objectives and characteristics of e-Science Grid infrastructures, without referring explicitly to Grid computing architecture or terminology. The chapter will conclude in the definition of a generic e-Science scenario as a set of characteristics and the specification of the assets of a corresponding e-Science infrastructure. The scenario as well as the assets represent the foundation of the work subsequently presented within this document.

2.1 E-Science in Particle Physics

The analyses and work presented in this document utilizes the *ALICE (A Large Ion Collider Experiment)* [1] experiment's e-Science Grid infrastructure as its main reference point. The ALICE experiment is one of the four large experiments within the Large Hadron Collider (LHC) [31], with *ATLAS*

(A Toroidal LHC Apparatus) [8], *CMS* (Compact Muon Solenoid) [12] and *LHCb* (Large Hadron Collider beauty) [23] as the remaining three large experiments.

The LHC is a particle accelerator and collider ring with a circumference of approx. 27 kilometers [94]. It is located at the border of France and Switzerland near the city of Geneva in a tunnel more than 50 meters under the surface and is operated by the European Organization for Nuclear Research (CERN) [18], a physics research institute and international organization.

2.1.1 ALICE Experiment

The particle physics experiment *ALICE* is based on a particle detector within the LHC, able to detect and identify the results of proton and nuclei collisions, and is focused in particular on heavy-ion collisions. Figure 2.1 shows a sample of lead-ion collisions recorded by the detector.

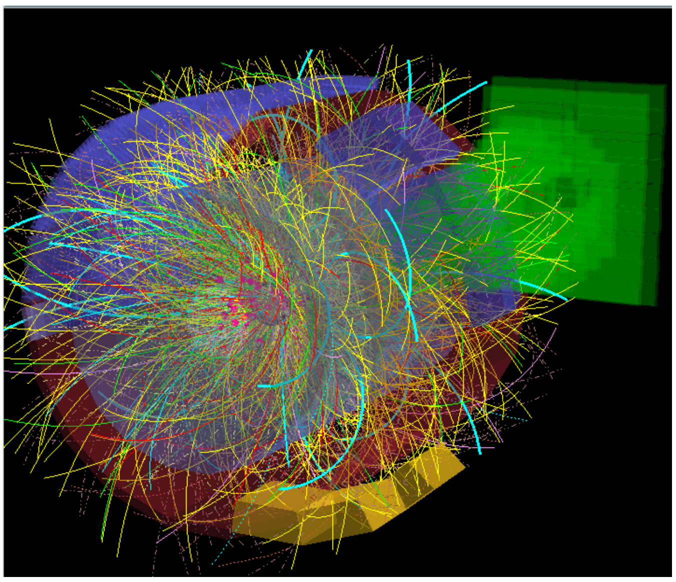


Figure 2.1: Events recorded by the ALICE experiment from the first lead-ion collisions in 2011. Source: [2]

Data acquisition The detector's data acquisition system generates raw data sets describing selected particle interactions, within the detector's measurement range and within a certain time frame, at a yearly data rate of more than 1 petabyte per year during LHC operation. These raw data sets are reconstructed in order to identify e.g. the concerned particles, their tracks, energy and lifetime. The reconstructed data is then analyzed in matters of confirmation of existing or formulation of new physics theories and models.

ALICE Collaboration The ALICE detector and all its infrastructure is operated by the ALICE Collaboration [2], which has approx. 1200 members from 132 participating institutes in 36 countries.

2.1.2 ALICE Grid Services

The *ALICE Grid Services* [3], a globally distributed open science and research cyberinfrastructure, are the main computing facility of the ALICE experiment. Its infrastructure is currently composed of more than 70 computing centers, provided by collaborating institutes located in over 30 countries, combining up to 50k CPU cores and 30 petabytes of data storage and is serving approx. 1000 active users within the ALICE Collaboration. The ALICE Collaboration as the operator of the ALICE detector as well as the ALICE Grid Services represents a *Virtual Organization* [79]. With the collaborating institutes providing storage and computational resources in their computing centers to the ALICE Grid Services, the ALICE VO includes both its computing environment's resource providers and its users.

Operational tasks The ALICE Grid Services are used for the three tasks of *simulation*, *reconstruction* and *analysis* of detector data [129, 126, 83]. The raw data generated by the ALICE detector's data acquisition system is directly stored on storage sub-systems attached to the ALICE Grid Services and consecutively replicated and processed with the Grid. The processing is organized and parallelized based on data sets, following the distinctive criterion of an *event* in the detector as the root cause for a certain set of data. All three tasks are conducted as both, general processing, as well as within single user's activities. Accordingly, the systems usage is classified into *scheduled processing* or passes organized centrally within the collaboration, and *user-driven processing*.

Service level The ALICE Grid Services provide both a Platform as a Service as well as a Software as a Service layer, whereas the latter is based on unified research software appliances. In essence, the infrastructure provides a global file system by unifying a distributed storage infrastructure and a centralized job submission portal, distributing computing tasks to its computing centers. The job submission is based on predefined task descriptions with no direct influence by or interaction with the submitting user during a job's runtime.

Usage Users are allowed to deliberately submit any kind of job, have to declare [92] though to use the infrastructure only in line with their research within the experiment. It is constituted by policy [47] that all data, be it produced by the detector or by data processing within the infrastructure's environment, can be accessed by every member of the collaboration, viz. by every user within the ALICE Grid Services.

Collaboration The supply and maintenance of resources by members of the ALICE collaboration, including the computing resources provided to the ALICE Grid Services by the institute's computing centers, are free of charge. The incentive for contribution is, next to the institute's participation in the experiment as such and the thereby assumed reputation, the access to the experiment's data and research for an institute's members.

2.1.3 Other Particle Physics Experiments

Although the four large LHC experiments ALICE, ATLAS, CMS and LHCb focus on different research aspects of particle physics and are based on distinct detectors and Grid environments, they were developed and built next to each other and share the same scenario [126, 127, 128, 131]: analogous to the ALICE Grid Services, the ATLAS, CMS and LHCb utilize each their own Grid-based e-Science infrastructure for simulation, reconstruction and analysis of respective detector data and are based on collaborations, representing both the Grid users and institutes providing computing resources. All infrastructures provide unified data layers and frameworks for submission of predefined jobs for scheduled and user-driven data processing. Their key figures are as follows: the ATLAS experiment and its collaboration [9] has approx. 3000 members, more than 1000 active Grid users [99], and over 130 participating computing centers [89]. The CMS experiment's collaboration [13, 14] has more than 3000 members, more than 1200 users of its Grid infrastructure and over 100 participating computing centers [68, 39]. The LHCb experiment, collaboration [32] more than 600 members from 17 countries.

Worldwide LHC Computing Grid The four collaborations of the LHC experiments represent Virtual Organizations united within the *Worldwide LHC Computing Grid (WLCG)* [35, 129], an umbrella organization established as a common Grid computing infrastructure, responsible for the coordination, operation and maintenance of the participating VO's computing centers, infrastructures and services.

Further examples of analogous e-Science scenarios in particle physics are the experiments *CBM* [11, 113], *PANDA* [33, 109, 113] and *SuperB* [40].

2.2 E-Science Examples Beyond Physics

With a multitude of existing e-Science infrastructures and according usage scenarios, some selected specifications and examples are presented as follows:

In [125], a study of biomedical research workflows and patterns with respect to a potential utilization of high-performance and Grid computing infrastructures is presented. The specified use cases are the simulation, preprocessing and analysis of scientific data, based on distributed computational infrastructure and unified cross-organizational access to distributed and independent data sources.

The *GEO (Global Earth Observation) Grid* [82] establishes unified data storage of and access to satellite imagery, geological archives and sensors of different organizations and collaborations. It further provides facilities for distributed simulation and analysis, e.g. in geological and ecological research and disaster mitigation [117].

The *e-BioGrid* [16] is a dutch e-Science infrastructure for life science research. It is based on a gateway for biomedical data analysis [119] connected to the national e-Science Grid infrastructure *SURFSara* [29]. The gateway provides unified access to computing facilities and software appliances for biomedical research. The e-Science infrastructure is utilized for simulation and analysis of e.g. biological samples, genomics and nanoscopy and medical image data. A further example of an e-Science infrastructure based on a gateway to an independent Grid infrastructure is the *Charité Grid Portal* [135], which has an analogous usage scenario in medical research.

2.3 An e-Science Scenario

Based on the e-Science approaches presented in the previous sections, the characteristics of a globally distributed e-Science cyberinfrastructure will be summarized as a generalized scenario as follows:

Collaboration within Virtual Organizations Independent research organizations and possibly single researchers collaborate within the organizational framework of a Virtual Organization. The collaboration presumably has an international basis and the utilized infrastructure combines internationally distributed facilities.

Unified data acquisition and distribution The acquisition or recording and unification of valued scientific data within a data framework. At the same time the framework arranges for data replication within a geographic and organizational distribution.

Unified distributed processing The consolidation of discrete processing facilities into one logical framework providing a unified coordination and handling of distributed processing. Via a dedicated service layer, researchers are able to conduct ***simulation***, ***preprocessing*** and ***analysis*** of scientific data via submission of predefined task descriptions, which are executed within the infrastructure without their attendance. They are thereby technically enabled and allowed to supply program code and data, and to use these in line with their research.

Synergy and collaboration The combination and unification of distributed knowledge and data as well as human and technological resources within a collaboration and a corresponding framework instead of concurrent or discrete operations and research.

Distributed unified access Cross-organizational unified access to distributed data and processing facilities.

2.4 Assets of an e-Science Infrastructure

In this section, assets of an e-Science VO with respect to its distributed research infrastructure will be presented. The section is based on a generalization and extension of the list of assets defined in the WLCG Risk Assessment [PUB8].

The first two identified assets of an e-Science VO concern the relationship of its members and participating organizations and the terms of their association.

Trust and collaboration The trust between members, collaborating infrastructures and organizations, external partners and funding agencies and the collaboration established on this trust are fundamental operational prerequisites and must be protected and maintained.

Reputation The opinion of the general public, funding agencies and members about an e-Science VO as an organization and its operation is a primary asset. Due to the mutual relation this includes also protecting the reputation of all members and participating organizations joined together within the technical framework of the e-Science VO.

Trust, collaboration and reputation represent the foundations of an e-Science infrastructure as an entity arranged for the interaction and cooperation of distributed components provided by multiple organizations and players. Any interference with operational assets as presented below is thereby assumed to have potential impact on these foundational assets.

Data integrity The accurate storage, lack of alteration and consistency of acquired and handled data within the infrastructure of an e-Science VO, including personal or meta data.

Data protection The sovereignty and control of access to data acquired and processed within the infrastructure of an e-Science VO, including personal or meta data, as well as data confidentiality and non-disclosure with respect to non-participating individuals. Depending on the valuation of an actual e-Science VO, this possibly also includes different levels of internal data protection within an e-Science infrastructure and protection.

Intellectual property and provenance Authorship and consequent intellectual property on scientific work and results, including techniques, procedures and methods utilized in the course of their achievement. This includes the ability to prove the provenance of scientific work and to enable reproducible findings.

Digital identities The integrity and availability as well as the absence of compromisation, exposure or forgery of any credentials, attributes or other information required for the correct identification, authentication and authorization of an e-Science infrastructure's users and entities.

Resources The integrity and availability of all entities that allow an e-Science VO to store data, meta data or any other information *(data resources)*, conduct computation and calculations *(computing resources)* and connect entities and establish communications *(network resources)*.

Service and infrastructure The integrity and availability of any software systems providing services relevant for the access, control, management, monitoring, processing or communication within an e-Science Grid environment.

2.5 Summary

A generic e-Science scenario has been defined in this chapter, based on the ALICE [1] experiment's e-Science Grid environment and other e-Science infrastructures and approaches.

In this scenario, research organizations collaborate within the organizational framework of a Virtual Organization (VO). This VO combines the distributed organizations and respective facilities and achieves their unification within one collaboration and a corresponding framework. On the basis of this framework, cross-organizational unified access to distributed data and processing infrastructure is provided to the VO's users. This enables the processing of scientific data via submission of predefined tasks (Grid jobs) subsequently executed within the distributed infrastructure, whereby the utilization of user-provided program code and data is facilitated.

Furthermore, assets with respect to the scenario have been specified, based on a generalization and extension of the existing definition of assets within the WLCG [PUB8]. Both the scenario and its assets represent the foundation of the work subsequently presented within this document and will be utilized in particular as the basis for a security analysis and definition of security objectives in Chapter 4.

"Architecture starts when you carefully put two bricks together. There it begins."

Ludwig Mies van der Rohe

3 E-Science Grid Architectures

In this chapter, a specification of the architecture of the ALICE Grid Services [3] and alternative e-Science Grid infrastructures will be presented, followed by the definition of an according generic e-Science Grid architecture.

In the first part of this chapter, the architecture and implementation of the ALICE Grid Services, and its central Grid middleware AliEn [4], will be analyzed, specifying the underlying infrastructure and layout, authentication and access control mechanisms, provided user interfaces and finally, the thereon-established Grid data and job layers. This detailed analysis will provide the basis for a vulnerability analysis of the ALICE Grid Services in Chapter 5.

Subsequently and in comparison to the ALICE Grid Services, architectures of alternative e-Science Grid infrastructures will be examined. Based on the comparison, a generic e-Science Grid architecture will be defined at the end of this chapter, which will constitute the reference Grid architecture for the subsequent chapters of this document.

3.1 ALICE Grid Services

The ALICE Grid Services are a globally distributed storage and computation Grid operated by the ALICE Collaboration [2] [1] as a Virtual Organization (VO). The central Grid middleware *AliEn (ALICE Environment)* [4, 111, 58] is an open source Grid framework, developed and maintained within the ALICE Collaboration. The ALICE Grid Services are embedded in the World-wide LHC Computing Grid (WLCG)[35] (see Section 2.1.3), the top tier association of the Grid activities of the LHC experiments. At the time of completion of this document, the ALICE Grid Services are based on more than 70 computing centers located in over 30 countries, combining up to 50k CPU cores and 30PB of storage [3], and are utilized as the main computing infrastructure by approx. 1000 active users within the ALICE Collaboration.

The central Grid middleware AliEn is written in Perl [34] and based on distributed services communicating via the Simple Object Access Protocol (SOAP) [102] while utilizing MySQL [26] databases and a Lightweight Directory Access Protocol (LDAP) [38] service as persistent data structure.

The central middleware establishes a globally unified *Grid layer*: a *Grid data layer* based on a central *file catalog* as a logical data structure and a distributed set of storage systems provided by the collaborating computing centers, and a *Grid job layer* providing a central *task queue* with a workload management system.

Beyond the ALICE Collaboration as the originator and main user, the ALICE Grid Service's architecture and its middleware AliEn are adopted and utilized also by the particle physics experiments CBM [11, 113] and PANDA [33, 109, 113].

Fabric The Grid fabric [77] of the ALICE Grid Services is most importantly based on a set of core computing systems running the so called *central services* located at CERN, and a multitude of distributed computing centers. A computing center, hereafter referred to as *Site*, provides each hundreds to thousands of *Worker Nodes (WNs)* as computing facilities and one or several storage systems, so called *Storage Elements (SEs)*. A Site's infrastructure is provided non-exclusively, viz. it must be assumed to be shared with further operations such as other VOs and their corresponding infrastructures or any other Site-specific utilization.

WNs on a Site are aggregated within the Site's resource management system and are accessible to the Grid Services via one or more batch system interfaces, so called *Computing Elements*. These Computing Elements, and thereby the Site's WNs, are connected to the ALICE Grid Services via ALICE-specific Site services, running on a dedicated Site-based computing system, called *VOBox*. The portal service to a Site, the AliEn *ClusterMonitor*, is running on a Site's VOBox and acts as an access gateway to a Computing Element and as a communication proxy for Grid jobs on WNs (see below).

All central services within the ALICE Grid Services as well as VOBoxes and WNs on Site-level are based on the Linux operating system.

3.1.1 Authentication and Access Control

This section will present a brief overview of the connectivity and authentication and authorization mechanisms in the ALICE Grid Services. More detailed specifications and analyses will be presented in the consecutive sections within this chapter and the remainder of this document.

Apart from mutual central service and Site-internal communication, as for instance between ClusterMonitor and WNs or WNs and SEs of the same Site, all communication of services and entities in the ALICE Grid Services is Internet-based. The authentication of services, Sites and Grid users within the ALICE Grid Services is established based on X.509 certificates [69] and X.509 proxy certificates [101, 132], utilizing the Globus Toolkit [76, 78]. The trust anchor of the certificate infrastructure is the *International Grid Trust Federation* [21], which maintains and provides access to root certificates of recognized certificate authorities, certificate revocation lists and policies as a so called *trust anchor distribution*. Any authentication via X.509 certificates or X.509 proxy certificates is verified based on this trust anchor.

All services, Sites and Grid users in the ALICE Grid Services are registered within records in a central LDAP directory service. A service entry specifies e.g. the service name, a uniform resource locator (URL) and its capabilities. Sites are represented by records specifying their capabilities and configuration settings and entries of all its Site services, e.g. the Computing Elements and SEs. Every recognized Grid user is listed with a record in the directory service, containing e.g. its username, real name, email address and an entry containing the distinguished name specified in the user's X.509 certificate.

Site services The services running on a VOBox are executed within the machine's operating system using a non-privileged user account. The communication between a VOBox and central services is mutually authenticated, based on an X.509 host certificate on the part of central services and X.509 proxy certificates for Site services. These proxy certificates are derived from the VOBox's X.509 host certificate, which is not directly accessible from ALICE Site services.

User access Access to the Grid layer by users is authenticated using X.509 proxy certificates derived from the user's X.509 certificate and the verification of a respective user record in the LDAP directory. Within the Grid layer, Grid users are referred to by the username listed within their LDAP record. All authorization and authorship of object entities within the Grid layer is based on these usernames, as well as group and role attributions.

A Grid user's system usage is limited by a quota mechanism with respect to Grid file entries and jobs. The accounted values are e.g. utilized Grid job run time and CPU cost, and the number of Grid files and utilized storage capacity.

3.1.2 User Interfaces

The ALICE Grid Services can be accessed via four different user interfaces: two command-line interfaces, a library providing Grid access from within the analysis framework as well as a graphical web interface.

The *AliEn Perl Shell* [48] is a command-line shell used by both Grid users and administrators, which connects directly to central or Site-based Perl services. A second command-line interface, the *AliEn Shell (aliensh)* [41] provides a rich user functionality and is the main interface for scientific users. Instead of a direct connection to Grid services, a dedicated central service, the *API service*, operate as an intermediary between its client interfaces and the Grid services. The initial authentication and authorization is based on the same mechanism as within the AliEn Perl Shell using the X.509 proxy certificate, while subsequently a session token is negotiated and used by client interface and service.

The *AliEn ROOT Interface* [41] provides access to the Grid from within the experiment's physics analysis software framework [6], utilized both by

Grid users and within Grid jobs as an interface for Grid file access. The connection to the Grid environment is established analogously to the AliEn Shell via the API service.

The *Alimonitor* [3] graphical web interface further provides user access as well as administrational controls and statistics based on monitoring data. The communication is mutually authenticated based on a Grid user's X.509 certificate and the web service's X.509 host certificate and is encrypted using the Transport Layer Security [70] protocol.

3.1.3 Grid Data Layer

The data layer of the ALICE Grid Services is based on a global virtual Grid file system established by one central file catalog [57, 50]. The file catalog as the logical data structure is stored in a set of MySQL databases and accessible via central services.

File Catalog Structure An entry in the file catalog is primarily referred to via a *Logical File Name (LFN)* entry within the POSIX-style Grid file system hierarchy, which consists of file entries as LFNs, directories and an additional data annotation system [49, 50]. Due to the these annotations, file entries can be associated and selected based on tags. Each LFN entry refers to a *Global Unique Identifier (GUID)* entry as an internal identifier of a Grid file, whereby several LFN entries can link to the same GUID entry.

The physical locations of Grid files are represented as *Physical File Name (PFN)* entries in the file catalog. A GUID entry refers to one or several PFN entries, representing each a copy of the same file. Thereby PFN entries can refer to either file entries on Site-provided SEs as trusted storage subsystems or to arbitrary other data locations within the range of supported protocols, such as SOAP or HTTP.

Access control Access to the file catalog is authorized based on file catalog entries and GUID and LFN entries have each their own POSIX-style user and group ownership and read, write and execute permissions. The access to PFN entries is regulated implicitly based on the authorization of a respective GUID entry.

Access to physical data entries on SEs is provided via a proprietary protocol [74]. Upon successful authorization of a request, a client receives an capability-based ticket, issued by a dedicated central service, and is consecutively able to access a physical file entry on an SE. Access to any not SE-based data is not authorized explicitly and consequently required to be directly accessible by a client. A detailed specification and analysis of the protocol will be presented in section 5.2.

Data management and transfers The AliEn Grid data layer allows to delegate the replication or movement of a physical data file from one SE to another via submission of *transfer requests*: a client can submit a textual transfer request specifying a GUID and respective PFNs to central services, and upon successful authorization, the request is placed in a central *transfer queue*. Queued requests are scheduled and propagated to dedicated *transfer agents*, which consecutively execute the physical replication or movement of the file data and modify the corresponding information of logical entries in the file catalog.

ALICE detector Within the Grid layer of the ALICE Grid Services, the ALICE detector is represented by a dedicated Grid user only. This user has prioritized access to the virtual Grid file system in order to allow the upload and registration of the detector's physics data. The data is physically distributed over Sites according to a tiered structure [97], based on performance characteristics, capacity and service level assurances.

3.1.4 Grid Job Layer

The computational layer of the ALICE Grid Services is based on the submission of Grid jobs as textual task specifications to a central task queue. The task queue is accessible via central services and managed by a central workload management system (WMS) [56]. The WMS processes and schedules submitted Grid jobs in the queue and initiates their propagation to Sites and consecutive execution on WNs.

The Grid job layer provides a Platform as a Service as well as a Software as a Service layer. The latter is established by a package management and distribution mechanism, providing automated on demand installation of Grid applications on WNs.

Grid job submission Grid jobs are specified by so called *JDL* entries (originating from *Job Description Language*) as a formatted textual specification derived from Condor ClassAds [15]. A Grid job is submitted to the central task queue either by reference to an according JDL file in the file catalog or directly as a textual JDL input. Within the JDL entry, references to Grid file entries in the file catalog are specified as LFNs. A submitted JDL entry specifies at least an LFN as the executable of the Grid job, which is later downloaded to the WN and executed as an operating system process.

JDL Once a JDL is submitted to the task queue, the WMS appends additional information according to the processing. Listing 3.1 shows a simplified example of the JDL entry of a Grid job, briefly explained as follows: the specification of the executable and arguments for the execution is followed by a listing of input files, which will be downloaded to the WN and can be accessed during the runtime of the job from within the WN process of the job as local files. Thereafter input data is specified, describing an XML [19] file containing a data collection that will not be downloaded to the WN and can be accessed remotely on an SE.

```
Executable = "/alice/user/a/auser/bin/train.sh";
Arguments = "1630 LHC11h";
InputFile =
   {"LF:/alice/user/a/auser/physics/train.root"};
InputDataCollection =
   {"LF:/alice/sim/LHC12a17d/esd.xml,nodownload"};
Split = "se";
Packages = {"VO_ALICE@AliRoot::v5-03-56-AN"};
OutputDir = "/alice/sim/LHC12/Electrons_PbPb_MC/";
OutputFiles = {"Events.root,Results.root,*.stat"};
User = "auser";
JDLPath = "/alice/user/a/auser/jdls/train.jdl";
```

Listing 3.1: "Example of a Grid job's JDL entry."

The subsequent splitting argument triggers the submitted job request to be split by the WMS into several sub jobs, based on the distribution of the specified input data over different SEs. The consecutive package requirements specify software packages to be installed on a WN before the job's execution. Finally, output directory and files are specified, which will be uploaded to SEs and registered accordingly in the file catalog.

JobAgent and Grid job execution Figure 3.1 shows an overview of the Grid job layer of the ALICE Grid Services, which is further explained as follows: instead of direct submission of Grid jobs to Sites' Computing Elements, so called *JobAgents (JAs)* are submitted as requests to Computing Elements by the AliEn central services. Once a JA request is executed on a WN, it becomes a temporary Grid service, connects to central services and requests Grid jobs as JDL entries for execution. This approach is generally known as the *pilot job model* or as *pull* [42, 58] paradigm. Due to this functionality, a Site's resource management system can be fully abstracted. WNs are dynamically integrated into AliEn Grid layer, with a JA receiving a fraction of a WN's computational capabilities. The submission of JA re-

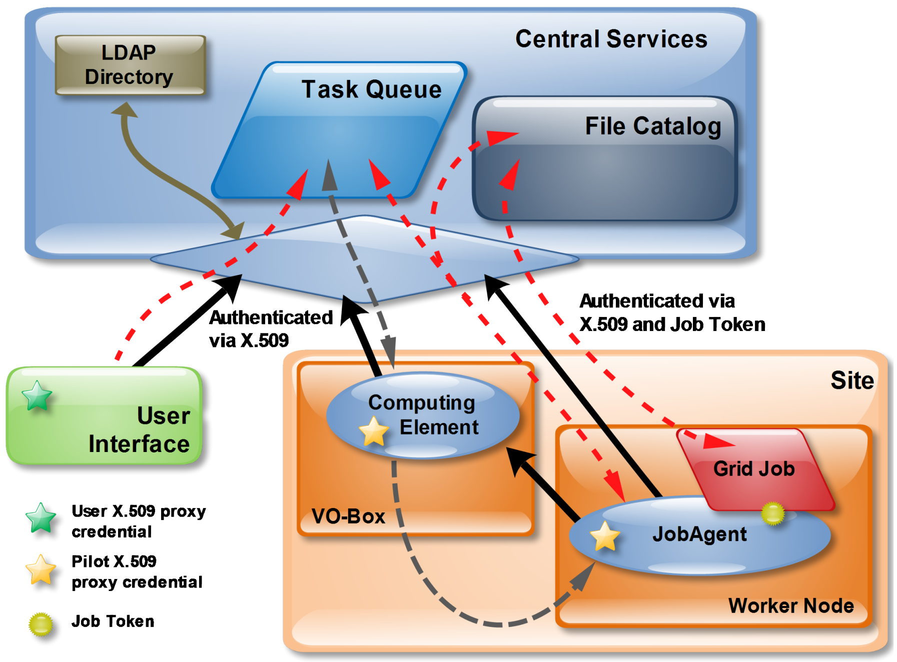


Figure 3.1: Pilot and Grid jobs in the ALICE Grid Services.

quests to a Site's Computing Element is authenticated and authorized by the ClusterMonitor service's X.509 proxy certificate on a VOBox. Thereby each submission includes a copy of the X.509 proxy certificate, utilized by the JA later on during its runtime. Once a JA request is sent to a WN for execution, its textual specification is deserialized, including the X.509 proxy certificate, and the JA is started as a service on the WN. The JA analyses its capabilities based on the WN's architecture and operating system with respect to Grid applications available for a later installation on the WN. By utilizing the X.509 proxy certificate for authentication and authorization, the JA connects to central services and the WMS and advertises its capabilities, including its time to live, and requests Grid jobs for execution. Once a match of a JA's capabilities and the requirements of a waiting Grid job in the task queue occurs, the WMS creates a password for later authentication of the Grid job, called a *job token*. The Grid job's JDL entry and the created job token is sent with the respective Grid job's JDL entry to the JA. The JA will perform potentially necessary installations of Grid applications based on the specifications within a JDL entry, and if successful will acknowledge the take-over of the Grid job to the WMS. The job token is subsequently used by the JA in order to authenticate and authorize requests during the Grid job's execution.

Once a Grid job is taken by a JA, the JA creates a local temporary working directory for the job, downloads its executable and input files according to the LFNs specified in the JDL entry and executes the job as an operating system process on the WN. During the Grid job's execution, the JA sends monitoring and resource utilization information to the WMS. The JA will not interfere with a Grid job's execution except the job exceeds its resource utilization limits, which will trigger the JA to terminate the job's processes. Upon successful execution of a Grid job, the JA will try to upload the output files to SEs and delete the local temporary working directory of the Grid job. Throughout its lifetime, a JA continuously processes Grid jobs one after another, including Grid job's submitted by different users. For performance reasons, the registration of output files in the file catalog is finally processed by a central service for every successfully executed job.

The file catalog and SE access concerning executable and input files before a job's execution, as well as the file access for specified output files, is thereby authenticated and authorized based on the job token, allowing the JA to impersonate as the Grid job submitter. Similarly, a running Grid job is able to access the file catalog using the job token via the ROOT AliEn Interface (see Section 3.1.2), where it is used to authenticate and authorize request to the central API service, again based on impersonation of the Grid job submitter. A running Grid job as a plain operating system process on the WN is further able to utilize the network and Internet connection of the WN, e.g. in order to retrieve or send data to remote entities.

3.2 Alternative Architectures

Within the following sections, the architecture of the ALICE Grid Services, as presented in the previous section, will be compared to alternative architectures of other e-Science Grid infrastructures and projects.

3.2.1 LHC Experiments

Within their association in the WLCG project (see Section 2.1.3), the Grid environments of the four big LHC experiments, *ALICE*, *ATLAS*, *CMS* and *LHCb*, are based on a common collaboration of computing centers as their Sites and share the same architecture concerning fabric and tiers: each of the four VOs uses Site-provided SEs as storage sub-systems, Site WNs accessible via Computing Elements and dedicated VOBoxes for VO-specific Site services (see Section 3.1). The common authentication mechanisms for service access are X.509 certificates and X.509 proxy certificates for all four infrastructures. Also all four infrastructures are based on the pilot job approach, while access to Sites' Computing Elements is granted via a dedicated X.509 proxy certificate allowing to submit pilot jobs to Sites' resource management systems (see also Section 3.1.4) [103, 39, 62]. Within the Grid infrastructures of ATLAS, CMS and LHCb, the data access to SEs is authenticated and authorized based on Grid user's X.509 proxy certificates [96], in contrast to the proprietary SE access protocol in ALICE (see Section 3.1.3).

The four Grid infrastructures are finally based on different central Grid middlewares, and consequently different central Grid services. Therefore, the Grid architectures of the VOs of ATLAS, CMS and LHCb are each shortly analyzed and compared to the architecture of the ALICE Grid Services within the following paragraphs:

ATLAS The Grid infrastructure of the ATLAS experiment is based on the distributed data management system *Don Quixote 2 (DQ2)* [55] and a Grid job layer provided by *Distributed Production and Distributed Analysis System for ATLAS (PanDA)* [98, 103].

The Grid data layer is established by a set of logical data repositories through the *Don Quixote 2 (DQ2)* [55] central service. The data layer uses *Datasets* and *Containers* as logical aggregations [81]: a Dataset represents a logical collection of Grid files, e.g. due to a common signification of respective data, and Containers are logical collections of Datasets, while e.g. different Datasets may refer to the same Grid file. The central catalogs contain all Container and Dataset information and Grid file entries. A file entry is based on LFN, GUID and an SE name as a reference to the file's physical location. The central catalogs contain no physical file location or PFN, while Site services provide local replica catalogs, mapping the logical file information to physical file entries on a Site's SEs.

Data transfers are based on a central subscription catalog [55, 134], to which transfer requests are submitted. Executed by dedicated Site-based services, data transfers operate physically on file entries yet are logically organized based on Dataset subscriptions.

Within the Grid job layer, PanDA is based on a single central queue for Grid jobs and provides interfaces for job submission and management. PanDA triggers on-demand transfers of Datasets to a Site via DQ2 upon necessity due to Grid jobs scheduled to be executed at a respective Site. Necessary Grid file data is then copied to the working directory of Grid job before its startup. Once a Grid job finishes, output files are copied from the working directory to a local Site SE and registered asynchronously [100, 72], authenticated and authorized based on the pilot job's X.509 proxy certificate [103].

Grid Applications utilizable within Grid jobs are either provided statically via Site-specific shared file systems or retrieved via the CernVM-FS network file system [89].

CMS The Grid infrastructure of the CMS experiment is based on a Grid data layer provided by the *Dataset Bookkeeping Service* and the *Data Location Service* [39], and a Grid job layer established by the *CMS Remote Analysis*

Builder [39, 68].

The CMS Grid data layer is based on a cascaded set of logical file catalogs, with one global file catalog provided as a central DBS instance and further Dataset Bookkeeping Service (DBS) instances or other logical file catalogs existing as local Site-based services. The Data Location Service offers look-ups of file entries stored within the DBS instances [73], with the central DBS instance containing only logical file data referring to Site entries as their physical data location [73]. The physical file data location is stored within the Site-based catalogs.

A data management service called *Phedex* enables the submission of transfer requests, providing the replication or movement of data. The mechanism operates on *datasets* as aggregations of logical files, with physical transfers being executed referring to file entries [39].

The CMS Grid job layer established by the CMS Remote Analysis Builder provides one central task queue, facilitating unified submission of Grid jobs requests. Grid jobs are scheduled and split according to input data proximity, while their propagation to Sites is based on glideInWMS [118]. Grid job output files are uploaded only to local SEs based on the same Site, and are registered logically on the Site's local DBS instance. The logical information is then propagated to the global DBS asynchronously [43].

Grid applications for Grid jobs are stored on a Site-based shared file systems, viz. all Grid applications are pre-installed with respect to a Grid job execution [39].

LHCb The central Grid middleware of the LHCb Grid infrastructure is *DIRAC (Distributed Infrastructure with Remote Agent Control)* [42], a central Grid middleware establishing both a Grid data and Grid job layer.

The *DIRAC File Catalog* consists of a set of central catalogs, providing logical information on Grid files and their physical replicas as well as additional meta data [120, 62]. File entries in the catalogs are based on LFN, GUID and PFN entries, specifying all logical data information centrally. Data management is provided by a central transfer queue, called the *Request Management System*, which receives textual transfer requests and propagates these to distributed transfer agents.

The DIRAC Grid job layer provides a central task queue, an integrated workload management system and its own pilot job implementation [64].

3.2.2 An Example in Biomedical Research

In [119], a biomedical research Grid portal is presented. Consistently with the WLCG scenario, the portal refers to an underlying Grid infrastructure providing SEs and WNs accessible via Computing Elements. Within the Grid data layer one central logical Grid file catalog with a file-based data management provides a unified Grid file system for its users. Users can further submit Grid jobs to a centralized queue, which are consequently propagated to WNs utilizing the pilot job model.

3.3 An e-Science Grid Architecture

Within this section, a generic Grid architecture will be defined, based on the commonalities of the ALICE Grid Services and its AliEn Grid middleware and the presented alternative architectures. The generic architecture will provide a reference framework and a standardized terminology for the remainder of this document, and will be hereafter referred to as ***e-Science Grid architecture***. Within this section, defined and hereinafter utilized terms will be indicated as italic bold text.

Overview and Fabric A distributed Grid layer is established based on central components and services, hereafter referred to as ***Central Grid Services***, and computing centers as resource providers, hereafter referred to as ***Sites***. A ***Virtual Organization (VO)*** or e-Science VO, as the operator of a Grid layer, provides and maintains the Central Grid Services. Computing centers as Sites provide storage and computing infrastructure as ***Storage Elements (SEs)*** and ***Worker Nodes (WNs)***. All services and entities are assumed to be based on sub-systems with discretionary access controlled Linux operating systems or analogous Unix-like operating systems.

The different services and entities are connected via private, shared and public networks, whereas the connections between clients and central Grid Services and between Central Grid Services and Site-based Grid services is by default assumed to be Internet-based.

Access control Access within the Grid layer is authenticated using X.509

certificates or derived X.509 proxy certificates and a set of trusted root certificates. Authorization is based on records provided and maintained via dedicated directories or services.

Grid data layer Within the Grid data layer, one or several ***Grid File Catalogs*** establish unified virtual Grid file systems as logical structures. A logical file entry in a Grid file system is referred to as a ***Logical File Name (LFN)*** entry. An LFN entry is either correlated to a ***Physical File Name (PFN)*** entry, specifying a physical file location on an SE or external resources, or to logical entries in another Grid File Catalog, while the latter accounts for potentially cascaded Grid File Catalogs. Further, an LFN can be linked to several PFNs in a *1 to n* relation as a file stored in *n* physical replicas. The authorization of access to the Grid file system is based on ownership and permission attributes of the logical Grid File Catalog entries.

In order to enable a delegated replication or movement of physical data, ***Grid transfers*** can be submitted as predefined textual specifications to a ***Grid Transfer Queue***. Grid transfers are submitted to and executed by distributed ***Grid Transfer Agents***, referring to the PFN entries of respective file entries in the Grid File Catalog.

Grid job layer Within the Grid layer, ***Grid jobs*** can be submitted by Grid users as predefined textual specifications to a ***Grid Task Queue***, where they are validated and processed. The processing may include transformations of an initial job submission, such as the splitting into multiple job requests, according to pre-defined rule sets.

The Grid Task Queue provides workload management functionality and maintains the scheduling and matchmaking of Grid jobs and WNs.

Site WNs are dynamically integrated into the Grid job layer by VO-specific services referred to as ***Grid Job Agents***, which receive a fraction of a WN's computational capabilities for a limited time, establish a connection to its Grid infrastructure and advertise the provided capabilities. Due to Grid Job Agents, previously unknown WNs are dynamically integrated into the virtual Grid layer for the lifetime of the respective Grid Job Agent. Grid Job Agents receive Grid job requests from a Grid Task Queue and maintain their execution as operating system processes on the WN. Therefore, the Grid job requests sent to Grid Job Agents must be self-contained statements, specifying explicitly the executable as well as necessary files, preconditions and target locations for their output. The Grid Job Agent is conceptionally assumed to execute a Grid job on behalf of a respective Grid job submitter or Grid user, and will erase any remaining data of the job, after its runtime. Furthermore, WNs are assumed to have a direct connection to the Internet that can be utilized by a Grid job. A Site's resource management is assumed to be encapsulated by a Site-service, provided and managed by the VO, referred to as a ***Grid Site Agent***. The Grid Site Agent is running on a dedicated Site-based computing system, referred to as the ***VOBoxes***, and maintains the submission of Grid Job Agent requests to a Site's resource management system.

Software appliances Using a predefined scope of available software applications, hereafter referred to as ***Grid Applications***, Grid job requests can specify software requirements and consequently presume their availability on a WN during execution. These Grid Applications are either statically available on a Site or WN, or are installed on a WN on demand before a respective Grid job's execution. The on-demand installation is assumed to be triggered and processed by a Grid Job Agent. The necessary packages are available via a data repository accessible within the Grid layer, e.g. the Grid file catalog.

User interfaces Grid users can connect to the Grid layer via ***Grid Client Interfaces***, providing access to the Grid file systems and allowing to submit and manage Grid jobs. Grid users are technically unrestricted to upload any data, including program code as data, to the Grid file system or to register data provided by external sources. They are equally free to specify the utilization or execution of any Grid file system entries within Grid jobs, provided that accesses to these entities are authorized.

3.4 Summary

A generic e-Science Grid architecture has been defined in this chapter, based on the comparison of the specified architecture of the ALICE Grid Services and other existing and instantiated e-Science Grid infrastructures.

According to the defined architecture, a virtual Grid layer is established by logical abstraction and within the framework of a VO, connecting a multitude of sub-systems as services and clients via uncontrolled networks, including the Internet. Collaborating institutes as Sites provide computing and storage infrastructure as Worker Nodes and Storage Elements, which are logically unified within a set of Central Grid Services, most importantly providing Grid File Catalogs and Grid Task Queues. Grid users are provided with access to virtual Grid file systems, established by Grid File Catalogs and Site-provided Storage Elements as physical storage devices. Further on, Grid users are provided with access to computational resources via the submission of tasks as predefined textual requests, hereinafter referred to as Grid jobs, to Grid Task Queues. These are subsequently distributed to Site-provided Worker Nodes and executed without a user's interference or attention.

This architecture will be utilized hereinafter as the reference architecture for e-Science Grids.

"We can only see a short distance ahead, but we can see plenty there that needs to be done."

Alan Turing

4 Security Analysis of e-Science Grids

In this chapter a security analysis of an e-Science Grid infrastructure will be presented, referring to an instantiation of the previously defined e-Science scenario (see Section 2.3) and e-Science Grid architecture (see Section 3.3).

After a discussion of related work, the involved players and security characteristics of an e-Science Grid as well as potential attacker motives will be analyzed and specified. Taking further into account the defined assets (see Section 2.4) of the e-Science scenario, the analysis will conclude with the definition of security objectives as general assessment criteria and the security demand necessary to be achieved.

Earlier versions of parts of this chapter have been presented in [PUB5, PUB7, PUB9, PUB10].

4.1 Related Work

Within this section, selected related work concerning Grid security analyses and the identification and specification of security issues and objectives will be presented.

In [78], Grid security requirements are assessed to potentially include *authentication*, *access control*, *integrity*, *privacy* and *non-repudiation*. Furthermore, the following constraints are presented: *single sign-on*, *protection of user credentials*, *interoperability with local security solutions*, *exportability* of code and a consequent infeasibility of ordinary encryption, *uniform credentials/certification infrastructure*, *support for secure group communication* as the information exchange of computational processes, and *support for multiple implementations* with respect to security policies. A *Grid security policy* is defined, based on *trust domains* as *logical and administrative structures* governed by a local security policy, summarized as follows: a Grid environment is established on multiple trust domains with heterogeneous sets of users, resources and local security policies. Thereby interactions between different domains are required to be consistent with all affected local security policies. Subjects within the Grid environment have a global and a local representation requiring their mapping between the global and each local trust domain. While the authentication of a global subject and its subsequent mapping to a local subject is required to be equivalent to the local authentication of the respective local subject, access control occurs only on local scope based on local subjects. Finally, mutual authentication of interacting entities belonging to different trust domains is required, and processes may be delegated subsets of user permissions in order to act on their behalf.

In [61], security issues concerning Grid environments are identified as Grid user management and accounts, missing user accountability with respect to behavior and system usage, in particular due to the usage of user-provided code.

In [123], security issues concerning on-demand Grids with large numbers of unknown users are specified as follows: *secure application deployment*, *worker node sandboxing*, *middleware separation*, *secure service deployment* and *secure workflow execution*. Furthermore in [121, 122], a security analysis concerning on-demand Grid and cluster computing identifies *authentication*, *authorization*, *delegation*, *confidentiality*, *secure communications*, *data availability* and *auditing* as primary security challenges. A defined scenario considers three types of actors, which are *resource providers*, *solution producers* and *users*. Solution producers supply software solutions and data to their users and utilize the resource provider's infrastructure for user-requested processing. The utilization of third party code and the possible requirement of a solution producer's root access to a resource provider's infrastructure are pointed out as particular security issues in on-demand Grid and cluster computing. Moreover, a dissolved differentiation of internal and external attackers is stated based on potential masquerading of external attackers, the possible use of third party code as well as possible collusion of insiders and external attackers. Three levels of security and trust requirements are defined for on-demand Grid and cluster computing environments. Thereby users are always required to trust their solution producer, while lateral or side by side operating users and solution producers require no mutual trust, sharing a resource provider's infrastructure. In the first trust level, a resource provider is required to trust the solution producers, concerning the non-malicious usage of their environment, with virtualization on resource-provider-side as a presented approach for solution. Also, in the first two levels users as well as solution producers are required to trust resource providers to correctly protect data and processing, which is assumed to be overcome by integration of trusted computing platforms.

In [66], a taxonomy of Grid computing security issues is presented, defining the three initial categories of *host level*, *architecture level* and *credential level*. The first category is further divided into *data protection* and *job starvation*, the second one into *information security*, *policy mapping* and *denial of service*. Concerning the first category, the authors briefly appoint different issues with respect to the exposure of host systems to data and program code from a priori unknown or untrustable entities and individuals. Virtualization and sandboxing mechanisms are presented as examples of potential approaches to these issues. Concerning information security, the category is subdivided into *secure communication*, *authentication*, and *single sign-on and delegation*.

4.2 Players and Relationships

According to the defined e-Science scenario (see Section 2.3) and architecture (see Section 3.3), an e-Science Grid is based on the interactions of three basic groups of players: first, the Virtual Organization (VO) operators managing the VO's Grid layer and maintaining its *Central Grid Services*. Second, the collaborating Sites as providers of storage and computing resources. And third, the VO's *Grid users*, utilizing the Grid layer. These three groups of players represent different entities with respect to domains of control [75] from a technical perspective, while the group of Sites must be further itemized into each an own domain of control per Site. A VO can be embedded

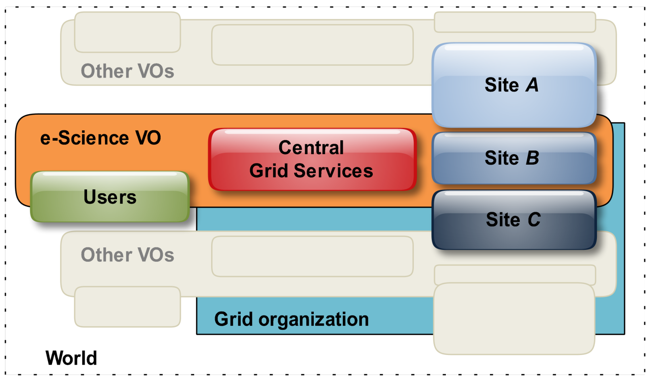


Figure 4.1: e-Science Grid VO, entities and relations.

within umbrella organization (e.g. the WLCG) representing several VOs with potentially common collaborating research institutions as Sites. In general, Sites are assumed to provide their resources in a non-exclusive and shared manner, viz. allowing any other VOs, organizations or user-groups to access their resources. Accordingly, next to the three player types within a VO, there are other VOs or organizations as adjacent or co-users of the shared infrastructure. These represent an imaginary fourth group of players, hereafter referred to as *third parties*. An example of the relations among the three types of players of VO, Sites and users as well as the VO's Central Grid Services, an umbrella Grid organization and third parties as other VOs are illustrated in Figure 4.1 as different overlapping layers.

4.3 Security Characteristics

In the following paragraphs, general security characteristics derived from the e-Science Grid architecture will be presented and emphasized.

The connection of distributed systems and services within an e-Science Grid includes the utilization of the Internet as a public network infrastructure or other shared and uncontrolled network infrastructures, leading to the following first characteristic:

Characteristic 4.1 ***Public communication environment*** Entities communicate within a public and a priori insecure environment.

The physical control, administration and sovereignty of Site infrastructure as well as local scheduling decisions remain at the Site level and a Site is free to share its facilities also with other parties (see Figure 4.1). Accordingly the following characteristics are stated:

Characteristic 4.2 ***Discretionary control of providers*** Sites as resource providers have the sovereignty and control over their infrastructure due to both physical and administrative access.

Characteristic 4.3 ***Discretionary resource sharing*** The provision and sharing of resources follows a Site's decisions and mechanisms as a provider and is only regulated via service level agreements.

Characteristic 4.4 ***Non-exclusive resources*** A Site's resources are not provided exclusively to an e-Science Grid and as such are shared with other consumers or users. Consequently interference with other consumers in the course of the utilization of resources must be taken into consideration.

An e-Science Grid provides its users with access to WNs at its Sites based on a Software as a Service or Platform as a Service layer, while users are permitted to submit and thereby execute arbitrary program code and data. Further, the payload submitted to WNs is assumed to be allowed to connect to arbitrary locations via private, shared or public (Internet) networks, e.g. in order to retrieve data or program code in the course of a Grid job's execution. In line with this discretionary usage, users are assumed to be obliged merely by policy to utilize the infrastructure only for the purpose of the approved research. E.g. the "Grid Acceptable Use Policy" [92] applied within the WLCG as well as the "OSG VO Acceptable Use Policy" [108] require any Grid user's activity to be in line with the goals and objectives of the respective VO. These aspects are summarized by the following characteristic:

Characteristic 4.5 ***Discretionary usage*** Grid users are allowed to introduce arbitrary data and program code into the Grid layer as well as request and initiate their utilization or execution within Grid jobs. Grid jobs are technically enabled to utilize Internet or other network access on a WN in a discretionary manner.

A Grid user can utilize the Grid Client Interfaces to connect to the Grid layer from any personal computing device, whereby a user's control over a respective device or system as well as the device's or system's integrity cannot be assured, which is expressed by the following characteristic:

Characteristic 4.6 ***Uncontrolled client environment*** An e-Science VO has no control of the Grid client environments nor can a Grid user's sovereignty and control over the respective computing system be taken for granted.

Due to the e-Science VO's nature based on international collaboration of different independent organizations and institutions, mutual relations between participating entities must be respected. This is expressed by the following last characteristic:

Characteristic 4.7 ***Cross-organizational access*** An e-Science Grid represents a conjunction or intersection of independent organizations or institutions as legal entities in different countries or states. Hence, legal concerns, like liability or due diligence, apply not only within the e-Science Grid as such but as well mutually between each of the interacting entities.

4.4 Attacker Motives and Targets

Potential attackers' motives and targets concerning an e-Science Grid must be considered as a multidimensional issue due to an e-Science Grid's cross-organizational capacity (see Characteristic 4.7), its shared underlying infrastructure and the direct and indirect interconnections with other parties (see Characteristic 4.3). This has e.g. particular impact on the differentiation of *insiders* and *outsiders*, due to the overlapping and intersection of system layers in an e-Science Grid. As an approach for understanding potential adversaries' view, attack motives and targets with respect to an e-Science Grid are briefly classified as follows:

Target 4.1 ***VO as primary target*** A motive targeting the primary mission of an e-Science VO as an intended interference with the operation or an attempt to hinder a VO from carrying out its tasks is referred to as an attack with the *VO as the primary target*.

Examples of such a motive could be to delete, alter or forge data or obtain read access, to generally damage or influence services or to manipulate a data taking process, in case of existing respective data sources.

Target 4.2 ***Secondary target*** A motive affecting though not directly targeting at the primary mission of an e-Science VO is referred to as an attack with a *secondary target*. It can be further distinguished by targeting at either *individuals* with a position or function within the VO including the VO's users, a *Site* within the VO, or any other *third party* the VO's Grid layer or operation is correlated with.

Assumed examples for a secondary target are an attacker trying to hinder VO members from doing their tasks or a potential usage of the system as an interface or intermediary entity to access infrastructure or operations of Sites or other VOs via a WN.

Target 4.3 ***System as a target*** Any otherwise motivated mission, e.g. demand for wisdom or vandalism or exploitation of the Grid infrastructure for any further agenda is referred to as an attack with the *system as a target*.

Here, an attack's appearance could be equal or similar to the case of the VO as the primary target or a secondary target, yet the actual motive would be different. Considered examples of this case are the incorporation of WNs as tools for attacks on any third parties via the Internet, or as well the potentiality of simply the lust for destruction.

4.5 Security Objectives

Within this section, security objectives for an e-Science Grid will be defined, derived from the combination of the previously specified assets (see Section 2.4) and the concerned players and their relationships (see Section 4.2), the specified characteristics (see Section 4.3) and potential attackers' targets (see Section 4.4). The objectives are considered to represent the central protection goals of an e-Science Grid infrastructure and will be utilized hereinafter as general assessment criteria.

The first objective concerns the assured integrity and attribution of data, in particular concerning the discretionary control of Sites (see Characteristic 4.2), the cross-organization level of access (see Characteristic 4.7) and the discretionary usage (see Characteristic 4.5):

Objective 4.1 ***Data integrity, authenticity and authorship*** The integrity and authenticity of all data must be ensured and potential violations are required to be identifiable. The authorship and provenance of any data must be recorded, providing verifiable and non-repudiable information.

Illegitimate system access signifies potential damage to any of the defined assets and must therefore be prevented. Considering potential attackers' motives (see Section 4.4) and consequently attacks from both outside and inside the central infrastructure or its sub-systems, an e-Science Grid's protection must respect attacks on or exploitation of the Grid itself (see Target 4.1 and Target 4.3) as well as attacks or side-effects of attacks on its Sites, users or a third party (see Target 4.2). A further concern is the protection against proliferation of attacks, such as illegitimate access or attacks on infrastructure or sub-systems spreading 'upwards' or across the Grid layer. Furthermore, users' utilization of the e-Science Grid is required to be restrictable, for instance by means of malicious or exceeding performance interference. Finally, an e-Science Grid should be able to adapt to the availability, and potential service outages, of its Sites and any other relevant infrastructure wherever possible.

Objective 4.2 ***System integrity*** An e-Science Grid and its infrastructure must be accessible only to authorized members of the respective Virtual Organization and be protected from any illegitimate access or alteration.

Objective 4.3 ***Availability*** An e-Science Grid and its infrastructure must be protected against being generally affected or struck by Site-based control and illegitimate access by or attacks on Sites. Concerning according malfunctions and outages, a Grid layer is required to maintain operability without affected sub-systems and Sites, and provide failover mechanisms and protection against data loss.

With respect to potential attackers' motives (see Section 4.4) and due to an e-Science Grid's characteristics of discretionary usage (see Characteristic 4.5), cross-organizational access (see Characteristic 4.7) and discretionary control of Site-based sub-systems (see Characteristic 4.2), responsibilities and liabilities of the involved parties become a further central security concern. As a reference point, Grid policies applied within the Worldwide LHC Computing Grid (WLCG) [35], the European Grid Infrastructure (EGI) [17] and the Open Science Grid (OSG) [28] are considered.

The *Grid Acceptable Use Policy* [92, 114, 108], as applied within WLCG, EGI and OSG, states a user's liability for consequences of misbehavior or security breaches. The *Grid Security Policy* [90, 115] further states the obligation of Grid users to protect their Grid credentials and their responsibility for consequences of misuse of their credentials.

The *Grid Security Traceability and Logging Policy* [91, 116] declares the following traceability requirement: *"The minimum level of traceability for Grid usage is to be able to identify the source of all actions [. . .] and the individual who initiated them."*

Hence, an initial job submission of a Grid user and all its consecutive modifications within the Grid layer must be each undeniably determinable. In reply to these concerns the following objective is defined:

Objective 4.4 ***Non-repudiation of Grid jobs*** A Grid job's authenticity and origin must be verifiable at any later stage, facilitating non-repudiation of a Grid user's initial job submission, consecutively applied modifications or transformations and its propagation for execution, even beyond the revocation of a Grid user's access and respective digital identities.

Data and meta information in an e-Science Grid must be generally assumed to be private and must therefore be protected from unauthorized access. Considering though the ALICE VO [1] as a reference point, it is constituted by policy [47] that all data and code must be publicly available to every member of the collaboration. Consequently any non-authorized access to the system represents a threat to data privacy, as within the system there are no further control mechanisms foreseen.

Nevertheless, data protection efforts need to reflect the condition that every member within the collaboration is technically capable of leaking any information or data without permission, and thus must be trusted not to do so. This relaxation of data privacy is not assumed, though, to be generally appropriate or applicable, resulting in the following objective as a lower limit for the herein-discussed general case of an e-Science Grid:

Objective 4.5 ***Confidentiality and data privacy*** Beyond access control and prevention of illegitimate system access, the storage, transport and processing of any e-Science Grid data or information must be protected against unauthorized information retrieval, such as copying, eavesdropping, sniffing or redirecting of data.

These five security objectives will be utilized hereinafter as general assessment criteria and are considered to represent the security demand necessary to be achieved within an e-Science Grid infrastructure.

4.6 Summary

In this chapter, the involved players and their relationships, the security characteristics as well as potential attack motives and targets with respect to the defined e-Science scenario (see Section 2.3) and Grid architecture (see Section 3.3) were analyzed and specified. Having further taken into account the defined assets (see Section 2.4) of the e-Science scenario, derived security objectives have been defined.

E-Science Grids are based on the interactions of three basic groups of players: first, the Virtual Organization (VO) operators managing the VO's Grid layer, second, the collaborating Sites as providers, and third, the VO's Grid users. These three groups of players represent different entities with respect to domains of control from a technical perspective, with the group of Sites being further itemized into an own domain of control per Site. An imaginary fourth group of players are third parties like other VOs or organizations as adjacent or co-users of the shared infrastructure. Moreover, there are umbrella Grid organizations in which several VOs can be embedded in.

The security characteristics of an e-Science Grid have been defined as its public communication environment, in particular the Internet, the discretionary controlled and non-exclusive Site-provided resources, the discretionary usage by its users and uncontrolled client environments, and the cross-organizational access within the infrastructure.

Three types of attack motivations and targets have been identified as a VO and its mission as a primary target, secondary targets like individuals and involved or adjacent organisations and entities, and third, the system as a target as a demand for wisdom or vandalism or the exploitation of the technical infrastructure.

The security objectives that have been consequently defined concern data integrity and authenticity, system integrity, availability, non-repudiation of Grid job submission and processing, and confidentiality and data privacy. These security objectives represent an e-Science Grid's security demand necessary to be achieved and will be referred to in the remainder of this document as general assessment criteria.

"We cannot solve our problems with the same thinking we used when we created them."

Albert Einstein

5 Vulnerability Analysis of the ALICE Grid Services

In this chapter, selected vulnerabilities of the ALICE Grid Services and the central Grid middleware AliEn will be presented (see Section 3.1), referring to findings of the previous security analysis (see Chapter 4) and the defined security objectives (see Section 4.5) as the assessment criteria. The vulnerabilities will be presented in three sections, concerning access control of the Grid layer and the security of the Grid data and job layer.

The presented vulnerabilities will disclose the deficiencies of the approach of delegation by masquerade, and will thereby identify the Grid job delegation within an e-Science Grid infrastructure as a central concern that is necessary to be dealt with. The analysis of Grid job delegation will then be pursued in the subsequent two chapters. Further on, the findings of this chapter will be incorporated in the remainder of this document as lessons learned and problems that have to be overcome.

Earlier versions of parts of this chapter have been presented in [PUB3, PUB4, PUB5].

5.1 Grid Layer Access Control

The following four vulnerabilities concern the authentication and protection of network-based communication in the Grid layer, the Grid services' protection against external access and enforcement of authorization and access control.

Vulnerability 5.1 ***Bypassable client authentication*** Up to AliEn version 2.18, the communication of Perl-based services and clients using the SOAP protocol is not authenticated, allowing to use certain Grid layer functionality in the name of any existing Grid user. As a consequence, it is e.g. possible to submit arbitrary Grid jobs using any existing identities without any other restrictions than the impersonated Grid user's permissions. A thereby submitted Grid job cannot be distinguished from a legitimate Grid job and will have full Grid layer access in the name of the impersonated Grid user during its execution on a Worker Node (WN).

This vulnerability was proven in practice to be easily exploitable: the AliEn Shell client interface initially attempts to establish an authenticated connection to the Grid layer's central database system and terminates itself in case the connection could not be established. By (small) modifications of the interpreted Perl source code on client-side, the interface ignores the missing database connection and successfully connects to central services without any authentication of the request or the pretended username.

Vulnerability 5.2 ***Direct database connection*** Due to the direct database connection of clients, malicious usage of database access as well as SQL code injections via the AliEn client interfaces are feasible, leading to potential damages and alterations of data.

Different uses of exploitation of this attack were proven in practice with respect to the manipulation of data in the AliEn file catalog and task queue.

Vulnerability 5.3 ***Missing service authentication*** Up to AliEn version 2.18, the communication of any Perl-based services and clients via the SOAP protocol utilizes no authentication methods for services. This leaves the services vulnerable to spoofing attacks, viz. clients might be fooled to connect to and communicate with forged non-authentic services. This vulnerability equally applies to the central API service, the MySQL database services, the LDAP directory service as well as to Storage Elements (SEs) and supported external data protocols, such as HTTP and SOAP.

Vulnerability 5.4 ***Interceptable communication*** Up to AliEn version 2.18, the communication of any Perl-based services and clients via the SOAP protocol is not encrypted, which makes interception of these communications possible. Again, this vulnerability equally applies to the communication with the MySQL database service, with the LDAP directory service as well as data communication with SEs and supported external data protocols, such as HTTP and SOAP. Furthermore, the communication with the central API service is only encrypted for client-to-service requests, while service-to-client responses are transferred as plain text.

The missing protection of the communication of Perl-based services and clients via the SOAP protocol and the service-to-client communication of the API service facilitates the interception of the credentials (job tokens) utilized for authentication and authorization of Grid jobs and the credentials to access SEs.

The encryption of client-to-service requests to the API service is based on the Advanced Encryption Standard with the session token as the encryption key. The implementation allows a fully encrypted communication, which is though disabled by default for performance reasons.

In the course of the security analyses of this thesis, these four vulnerabilities were identified as major security threats and became thereupon approached in a first stage: in AliEn version 2.19, mutual authentication and encryption were introduced [PUB3] , yet only for SOAP-based connections to Central Grid services. The implementation uses the Apache [30] web server as a service front end with mod_ssl [24] and mod_gridsite [20] modules providing encrypted communication and mutual authentication via X.509 host certificates and X.509 proxy certificates. The direct database connection of clients and non-central services was replaced by an integration of the respective data exchange into the SOAP-based communication.

Vulnerability 5.5 ***Unprotected Grid user credentials*** The X.509 proxy certificates utilized for client authentication by the AliEn Perl Shell client, the AliEn Shell client and the AliEn ROOT interface are not protected against theft. Hence, arbitrary illegitimate utilization is possible during their period of validity. The credentials are stored as plain-text information in file entries in the operating system's temporary folder and are only protected by the file system's POSIX-permissions.

Accordingly, once obtained X.509 proxy certificates can be utilized to access the Grid layer from any other location. This facilitates identity-theft without any further restrictions than the credentials' validity, which is 12 hours by default. The characteristics of X.509 proxy certificates are further analyzed in Chapter 6.

5.2 Grid Data Layer Security

The vulnerabilities presented in the following apply to the Grid data layer of the AliEn middleware up to version 2.18. Solutions and counter-measures were presented in [PUB4, PUB5, PUB10], including their integration into AliEn version 2.19, as an improved Grid data access protocol and are hereinafter presented in Section 8.1.

In order to read or write a file on an SE accessible via the *XRootD* [36, 71] protocol a client has to connect to an AliEn central service, called *Authen*, beforehand and retrieve a so called *Access Envelope* [74], containing a capability-based ticket. An Access Envelope encloses the ticket together with a public-key signature of a checksum of the ticket, based on RSA signatures and SHA1 checksums. Ticket and signature are protected by a Blowfish symmetric-key encryption. The key used for the encryption is itself encrypted using a public-key encryption and placed next to the encrypted ticket. The Access Envelope's structure requires two pairs of keys to be used as follows: the Authen service signs the created ticket with its private key and encrypts the Access Envelope using an SE's public key. A SE receiving an Access Envelope in the course of a request decrypts the Access Envelope using its own private key and verifies the ticket inside using the Authen service's public key. The validity of the Access Envelope is limited by a creation and an expiration timestamp specified within the ticket.

The encryption of the Access Envelope enables a secret transmission of information from the issuer to the final recipient, though this functionality is actually avoided in AliEn. In order to permit informed decisions concerning the data retrieval and detailed logging on client side, and since the ticket contains no secret information with respect to the user or client, the access ticket is additionally passed on as plaintext.

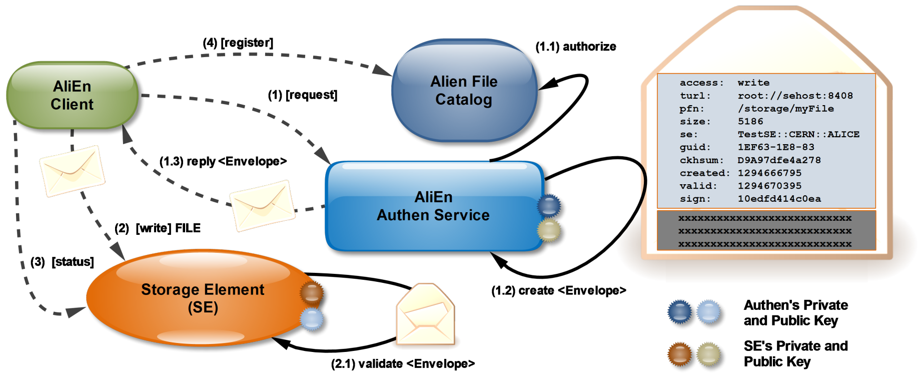
Figure 5.1 shows how the AliEn client interface retrieves and uses an Access Envelope in the course of a write access, which is explained as follows: The client sends a request specifying the Logical File Names (LFN) or Global Unique Identifier (GUID) and the SE, file size and checksum to the central Authen service. Upon successful authentication and authorization of the request the Authen service replies with a digitally signed ticket inside an encrypted Access Envelope. The client sends a write request including the Access Envelope and the file data to the SE via the XRootD protocol. The SE decrypts the Access Envelope and verifies the digital signature and content of the ticket in order to authenticate and authorize the access, and executes the write operation if applicable. After a successful write attempt the client requests the status of the written file in order to confirm the existence and size of the file. Finally, the client connects to the file catalog and registers the file specifying all necessary parameters.

The read access to a file on an SE has an analogue procedure, with an SE replying to the client the respective data of the file entry once the authorization of an according Access Envelope was successful.

As to deletion, normal Grid user's are only allowed to delete logical entries in the file catalog according to their permissions. The mechanism of Access Envelopes is though equally utilized in order to grant administrative users and dedicated services permissions to delete physical file entries on SEs.

The specified access protocol has several vulnerabilities, which are discussed as follows:

Vulnerability 5.6 ***Forgeable Grid files and attributions*** The creation of Access Envelopes is fully controlled by the Authen service and relies on the integrity of the service and the secrecy of the utilized keys, whereas potentially forged or illegitimately modified Access Envelopes cannot be



```
access:   write
turl:     root://sehost:8408
pfn:      /storage/myFile
size:     5186
se:       TestSE::CERN::ALICE
guid:     1EF63-1E8-83
ckhsum:   D9A97dfe4a278
created:  1294666795
valid:    1294670395
sign:     10edfd414c0ea
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Figure 5.1: Writing a file in AliEn v2.18 using the SE access protocol.

identified or discovered. Furthermore, all logical file entries and file entry attributions, such as LFN, owner and permissions, are maintained as simple value fields in the database of the file catalog. Again, this undermines identification or discovery of forged or illegitimately modified entries.

Consequently, everybody in possession of according privileges within central computing systems is able to modify or forge Grid file entries as a whole or Grid file attributions within the file catalog.

Grid users are further allowed to register arbitrary file entries in the file catalog:

Vulnerability 5.7 *Bypassable file ownership and permissions* The file registration in the file catalog is unverified concerning entry's Physical File Name (PFN), size and checksum, as well as ignoring any potentially existing references to a specified PFN. The missing cross-checking of PFNs therefore allows to bypass ownership and permissions of physical file entries by registration of already existing PFNs as new logical file entries in the file catalog.

By retrieving the PFN value of an existing file entry in the file catalog, a Grid user is able to register a new file entry specifying this existing PFN, which will result in a new file entry in the file catalog belonging to the user. As permissions and ownership within the file catalog apply only to LFN and GUID values, the Grid user will implicitly receive all permissions with respect to the physical file the PFN refers to, even if not entitled to write the original logical file entry.

Vulnerability 5.8 ***Race conditions and physical file overwriting*** Since the initial authorization in the course of an Access Envelope request is independent of the later file registration, race conditions concerning LFN or GUID collisions may occur. In the latter case, overwriting of newly created physical files is possible.

Consequently, once two clients attempt to create a new file with the same LFN concurrently only the client registering the file entry as first in the file catalog will be successful, even if the other client has requested an according Access Envelope before. Moreover it is possible to overwrite a physical file in the course of a file entry's creation: during the generation of an Access Envelope by the Authen service, the PFN of a new file entry is composed by SE specific parameters plus its GUID, while a specific GUID value can be optionally requested by a client. A user-specified GUID will be accepted, if no registered file entry with the respective GUID exists within the file catalog. Consequently, within the time frame of a physical file's creation on an SE and its logical registration in the file catalog it is possible to request write access to the same GUID and SE and to overwrite the already existing physical file. The loss of the original file data will thereby be only identified once the respective file entry will be accessed later on. This affects e.g. the output files of Grid jobs, within which the registration is deferred with respect to the physical files' creation on an SE (see Section 3.1.4).

Vulnerability 5.9 ***Client-trusted file identity and size*** The Authen service never examines the physical content of a file and relies on the client's unverified declaration of a file entry's checksum and size.

Accordingly the checksum and size entries within the file catalog are an unreliable and insignificant reference concerning a file entry's integrity and a user's file quota. Initially incorrect declarations cannot be distinguished from later modifications of physical file data and fraud against the system's resource accounting is facilitated.

Vulnerability 5.10 ***Untraceable alteration or loss of physical data*** Physical data of file entries can be lost, altered or data planted without later identification or discovery, as checksums are not enforced within file operations.

Accordingly, modified or corrupted data will be utilized within the system without notice. Only if multiple physical replicas of a file entry exist and are inconsistent or if data is identified to be unusable in the course of its processing, an alteration will be identified.

Vulnerability 5.11 ***Replay of Access Envelopes*** Access Envelopes are self-sufficient with respect to SE accesses and have no further binding to the Grid user or account in favor of which they were issued, yet are transmitted over an interceptable connection (see Section 5.1). Within the time frame of their validity, Access Envelopes can be utilized repeatedly as there is no state of their utilization considered.

In case of read accesses an Access Envelope therefore allows any holder to read a respective file entry on an SE during its validity.

Concerning write accesses, a flag in the Access Envelope indicates if the credential either allows to create a new physical file entry on an SE, while an SE is supposed to deny access, if a file with the specified PFN exists. Hence, a replay of an Access Envelope will have no effect on an existing physical file on SE. However, if by a malicious access a new file on an SE is created before the genuine request, the genuine client's request will fail and an illegitimate file will be existing.

In case of an access for deletion of a physical file on an SE, a replay will be without any effect. Yet again, the preempting of a deletion by illicit use of an Access Envelope must be assumed as a damaging event, as the point in time as well as the decision about the sending of the request would not be controlled by the genuine client.

5.3 Grid Job Layer Security

The Grid job layer in the AliEn Grid middleware is most importantly established on the approach of delegation by masquerade. Based on this approach, a Grid user submits Grid jobs to the VO's Grid services and the VO consecutively acts as a representative of its users. On Site-level, Grid jobs of all of the VO's users are executed in the name of the VO without any further distinction, at which the VO represents an intermediary between its users and Sites. In the course of a Grid job submission, a user performs an implicit delegation of its Grid layer privileges to the VO.

Consistent with the case of Grid file system entries, the attribution of Grid jobs in AliEn is maintained by textual entries, as e.g. the username. These entries are stored within or next to the job's JDL specification in the central task queue.

Vulnerability 5.12 ***Misuse and forgery of Grid jobs*** Due to the usage of delegation by masquerade, a Grid user's privileges can be deliberately and covertly misused at the level of central services or other processing entities, as all relevant information is controlled by the VO.

Hence, on Site-level there is no further insurance of a correct execution of a job beyond simple run time and resource utilization monitoring by the AliEn JobAgent (JA). Grid users have no possibility to prove non-conformity of the actions having taken place in their name and as such to protect the misuse of their digital identity, including potential forgery of Grid jobs and identity-theft. Due to the architecture of job submission and execution, it is virtually impossible to state the origin of potential security incidents, attacks, or misbehavior arising along or from a Grid job executed on a WN. A job could potentially be forged or altered at any processing point between user and WN or on the WN itself, while an alteration's origin cannot be identified and possibly not even the consequences or the alteration as such.

A JA on a WN executes all Grid jobs as received on behalf of the VO, in which the jobs are executed directly by the JA process and therefore run within the same user environment in the WN's operating system, leading to the following two vulnerabilities:

Vulnerability 5.13 ***Exposure of the JobAgent and missing Grid job isolation*** Grid jobs are not encapsulated or isolated, neither with respect to the JA service nor mutually, and are therefore able to tamper with the JA or other Grid jobs.

By malicious alteration of a JA, Grid jobs consecutively processed by the JA can be accessed and modified. A JA runs only one Grid job at a time and a job's working directory is scratched after the execution. Nevertheless, a job can also fork sub-processes that will remain active after the job's execution, which are able to alter consecutively executed jobs.

Vulnerability 5.14 ***Exposure of the JobAgent's credential*** As the JA's X.509 proxy credential has no protection itself (see Chapter 6) and the JA is neither protected against access from Grid jobs, the credential can be directly obtained or utilized by any Grid job.

The privileges of the credential allow to impersonate any of the VO's Grid users without restrictions. Consequently, a Grid job can start e.g. a Grid client interface on the WN, utilizing the X.509 proxy credential for authentication and authorization. Moreover, the credential could be sent out to other locations via Internet or be uploaded to the Grid file system and consecutively retrieved. With an initial validity of several hours (see Section 3.1.4), the limitation in time can thereby not be assumed as an actual protection against misuse.

Vulnerability 5.15 ***Impossible differentiation of Grid job origins*** Grid jobs originating from different users are not visible to Sites and a WN's operating system.

Hence, from a Site's perspective all Grid jobs of a VO correlate to one digital identity and it is not possible to differentiate Grid jobs in a transparent way. In case of security incidents or attacks, potential counter-measures can only affect all Grid jobs of a concerned VO.

5.4 Summary

Selected vulnerabilities of the ALICE Grid Services and the central Grid middleware AliEn have been presented in this chapter, concerning access control of the Grid layer and the security of the Grid data and job layer.

In summary, the ALICE Grid Services and its AliEn Grid middleware have strong architectural security deficiencies with respect to inbound and outbound protection and the authenticity of object entities and interactions within the established Grid layer. This is due to the reliance on unverifiable information concerning authenticity and authorship, and the utilization of delegation by masquerade with respect to Grid job processing and execution. The latter facilitates for instance unverifiable misuse of Grid user privileges and digital identities. Finally, privileged credentials on the WN are not protected against access or retrieval by executed Grid jobs. As a result, it is not possible to achieve the afore-defined security objectives (see Section 4.5).

"Your messages I hear, but faith has not been given"

Johann Wolfgang von Goethe

6 Analysis of Delegation and X.509 Proxy Certificates

In the previous chapter (see Chapter 5), the approach of delegation by masquerade as applied within the ALICE Grid Services implied fundamental deficiencies. However, the aspect of delegation represents a central concern within an e-Science Grid infrastructure that is necessary to be dealt with. Therefore, within this chapter X.509 proxy certificates [132] as a widely adopted mechanism for delegation of privileges will be analyzed in detail and their shortcomings and limitations will be specified.

First, the concept of delegation in the context of Grid computing will be investigated and two conceptual approaches will be distinguished, with the first approach being delegation by masquerade as applied within the ALICE Grid Services. With respect to the second advantageous approach of a transparent delegation, X.509 proxy certificates will be analyzed as an alleged mechanism for an according implementation. An examination of the thereby necessary handling and propagation of X.509 proxy certificates as delegation credentials within a Grid layer will identify several cardinal vulnerabilities.

Earlier versions of parts of this chapter have been presented in [PUB5, PUB7, PUB10].

6.1 Delegation in Grid Computing

In essence, delegation in Grid computing occurs once Grid jobs are handled by Grid services and finally executed on Worker Nodes (WNs). Referring to the defined e-Science scenario (see Section 2.3) and e-Science Grid architecture (see Section 3.3), Grid users submit job requests to a Grid layer, which are consecutively processed by Central Grid Services and are thereby in the sovereignty of the respective Virtual Organization (VO). Upon submission of these processed job requests to WNs on a Site, the VO acts as a delegatee, demanding the Grid jobs' execution on behalf of the respective Grid users.

Such a case of delegation is not limited to the case of Grid jobs and could also be necessary for other tasks requested by a user to be processed within the Grid, such as data transfer requests. To simplify matters, this chapter will focus on the delegation of Grid jobs, since delegations of other tasks are assumed to represent analogous or less complex scenarios.

An implementation of Grid job or task delegation can be classified according to two basic scenarios as follows: in the first scenario, a VO acts with respect to its Sites or resource and service providers as a proxy and representative of its Grid users. Grid jobs or tasks of all users are executed on WNs or service endpoints using a digital identity representing the respective VO as a whole. Accordingly, the VO masquerades as its users and the actual digital identities of submitters can only be resolved via an accounting or information provided by the VO. Hence, Grid jobs or tasks submitted by different Grid users cannot be distinguished directly at Site-level or on a WN. This approach is utilized e.g. in the ALICE Grid Service's middleware AliEn, with its deficiencies having been specified in Section 5.3.

In the second scenario, a Grid job or task is separately authenticated on Site- or provider-level using a credential on a WN or service endpoint before its execution, indicating and assuring the corresponding submitter's digital identity to the Site or provider. Moreover, the credentials are used on WNs or service endpoints to grant access to external services for a Grid job or task, e.g. to download and or upload data, with the intention to transparently authenticate and authorize such requests based on the submitter's identity.

The two concepts are defined as follows:

Definition 6.1 The delegated execution of a Grid job or task based on masquerading of users as unidentifiable submitters is referred to as a *masquerade-based delegation*.

Definition 6.2 Delegated execution of a Grid job or task facilitating the authentication of a submitter's identity and the consequent requests is referred to as a *transparent user delegation*.

The definitions are used hereafter in order to evaluate the utilization of X.509 proxy credentials and alternative mechanisms as approaches for transparent user delegation in comparison to the simpler masquerade-based delegation approach.

6.2 X.509 Proxy Certificates

The basic idea of X.509 proxy certificates [133, 132] is to provide single sign-on and delegation while not exposing the private key corresponding to a respective X.509 certificate [69], and to rather issue credentials of lower order, most importantly proposed and implemented in the *Globus Security Infrastructure* [78, 60].

An X.509 proxy certificate contains a public key (of a private/public key pair) signed with a private key corresponding to either an X.509 certificate or another X.509 proxy certificate. Next to the signed public key, the X.509 proxy certificate further contains the X.509 certificate or a respective chain of cascaded X.509 proxy certificates plus the X.509 certificate the chain is derived from, wherein one certificate asserts its successor by a signature. An X.509 proxy certificate and the corresponding private key together are hereafter referred to as a *proxy credential*.

Proxy credentials as applied for instance within the WLCG are based on unrestricted delegation, although the mechanism itself would allow further functionality, such as including specifications or limitations of its usage. Unrestricted delegation based on proxy credentials has though long-known cardinal security deficiencies [101], which have already been considered in the initial proposal for X.509 proxy certificates [133]. These deficiencies are defined and specified as limitations with respect to Grid computing as follows:

Limitation 6.1 ***Unrestricted delegation*** A unrestricted delegation is given wherever the delegation has no other dependency than in the dimension of time. This is the case if a proxy credential contains no explicitly specified limitation. Within its validity, a proxy credential then allows only for a full delegation to a delegatee, which consequently holds all privileges of the delegator.

Limitation 6.2 ***Unconditional delegation*** A proxy credential without further qualifications has neither any binding to a particular delegatee nor any context-sensitivity of its usage as any privilege is held as such.

In summary, a plain proxy credential neither specifies what is delegated nor to whom or in what context, and virtually represents a copy of an X.509 certificate which is identifiable as such and has a shortened validity in time with respect to the original X.509 certificate.

Any further limitation or qualification would though require additional mechanisms capable of interpreting and processing such qualifications of a proxy credential accordingly, and reject authentication or authorization in case of unknown specifications.

An extension to the X.509 proxy mechanism called *Virtual Organization Membership Service (VOMS)* [65] provides the possibility to apply authorization attributes to a proxy credential. Upon authentication using a valid proxy credential it is possible to request a VOMS service to provide a new derived proxy credential containing additional attributes which the respective proxy holder is entitled to use. These attributes can define e.g. a VO membership or roles and the mechanism prevents an attribute's value from being altered once set within a proxy credential.

Additional mechanisms like VOMS can provide further restrictions to the use of proxy credentials, if plain proxy credentials without extensions grant no critical permissions. However e.g. within the WLCG environment, VOMS extensions are primarily used to elevate the privileges of a proxy credential holder, e.g. to obtain the necessary role to run a pilot job (see Section 3.1.4). Thus, any holder of a plain proxy credential can request a new set of VOMS extensions within the range of privileges of the original certificate owner, and thereby escalate the level of privileges.

In [66], the necessity of additional specifications for X.509 proxy-based delegation is pointed out, while indicating policies and rules based on the Security Assertion Markup Language (SAML) [105] or the Extensible Access Control Markup Language (XACML) [104] as potential solutions.

Also in [53], the mismatch of requested processing and delegated privileges is identified, while referring to a least-privilege delegation as the optimal scenario. As an approach for solution the consolidation and unification of the languages utilized for the expression of jobs and resources and the consecutively necessary privilege specifications are proposed.

Limitation 6.3 ***Exposure to theft*** A proxy credential is by itself completely unprotected while being passed on within a distributed environment, such as an e-Science infrastructure.

Concerning this aspect of theft, a proxy credential is comparable to a plain security token. Without additional protection a proxy credential can be stolen at any location it is in use and must be expected to be accessible at least by persons with administrative or privileged access. Moreover, a proxy credential's validity of e.g. hours or days cannot be considered to successfully prohibit an exploitation by potential attackers.

Limitation 6.4 ***Multiple domain validity*** Since a X.509 certificate can be signed by more than one VO or any other institution, a corresponding proxy credential can as well be valid to access all these VO's or institution's infrastructures and resources as granted by the original X.509 certificate.

This presumes that different VOs and institutions equally recognize a proxy credential. Which could though at least be expected within umbrella Grid organizations and is e.g. the case within the WLCG, where it particularly concerns users with administrative functions in several VOs. Accordingly, the protection of proxy credentials must be an objective in all concerned VOs, and the organization maintaining the lowest security level will denote a lower boundary to all other organizations.

6.3 Propagation of X.509 Proxy Certificates

An adoption of X.509 proxy certificates (proxy credentials) for delegation implies introducing mechanisms for storage, handling and propagation of the credentials within a Grid layer. Within the following paragraphs, different mechanisms and approaches will be presented and analyzed, and two alternative approaches of usage and propagation of X.509 proxy certificates will be identified.

The credential management service MyProxy [51, 93] maintains proxy credentials after their initial upload and issues derived proxy credentials of lower validity to authorized entities on demand. Thus, a delegator uploads a mid-term proxy credential, having e.g. a validity of one month, to the service while specifying a password. A delegatee in possession of either the password or an equivalent (still) valid proxy credential can then request a short-term proxy credential, e.g. 24 hours, from the service. Moreover, a delegator can renew the proxy credential stored within the service, in order to enlarge the procedure beyond the initial time frame. Hence, in theory a proxy credential can be endlessly extended in time.

In [103], an implementation of a transparent user delegation in the PanDA (see Section 3.2) Grid job framework is presented: based on a MyProxy service, the Grid Client Interface uploads a Grid user's proxy credential to the service while specifying a random password. The password is sent by the client to the VO's Central Grid Services upon Grid job submission, where it is stored. Whenever a Grid job of the respective user is sent by the Central Grid Services to a WN for execution, the password is attached to the Grid job and used by the Grid Job Agent on the WN to retrieve the corresponding user's proxy credential from the MyProxy service.

Definition 6.3 An approach in which a VO holds keys with a one-to-one relation to its users' proxy credentials is hereafter referred to as an *indirect user proxy propagation*.

Within the Grid job framework DIRAC (see Section 3.2), transparent user delegation is implemented based on direct storage of proxy credentials within Central Grid Services [63, 95, 110]. The framework contains mechanisms analogous to the MyProxy service, wherein a Grid Client Interface uploads a Grid user's proxy credential to the Central Grid Services, which then issues derived proxy credentials of lower validity to its Grid services. Consequently, a Grid Job Agent on a WN receives a Grid user's proxy credential directly from the Central Grid Services once it receives a request to execute a Grid job.

Another example of this approach is the *Charité Grid Portal* [135]: there, Grid user proxy credentials are stored within the portal's services. Proxy credentials of lower validity are issued on demand in order to allow the portal to act in the name of the user while submitting requests to the Grid infrastructure below.

Definition 6.4 The approach of storage within a VO's sovereignty and direct transfer of user proxy credentials is hereafter referred to as a *direct user proxy propagation*.

Further, the *GEO Grid* [117] Portal middleware utilizes a MyProxy service in order to store and maintain proxy credentials of Grid users protected by passwords. Users send the respective passwords via an interface to the middleware in the course of system usage, which consecutively obtains and utilizes proxy credentials. The portal is based on a VO-controlled instance of a GAMA [54] service, which creates and manages proxy credentials, while users never directly obtain or control their proxy credentials and only utilize passwords to authenticate at the portal. Hence, this is assumed as an example of an escalated case of a VO's direct access to Grid user credentials.

6.4 X.509 Proxy-based Grid Job Delegation

Without comprehensive additional mechanisms the specified limitations of X.509 proxy certificates (proxy credentials) based on unrestricted delegation lead to fundamental weaknesses. An adoption as a mechanism for transparent user delegation introduces severe security vulnerabilities, which are defined and specified as follows:

Vulnerability 6.1 ***Unverifiable correlation of assignment and delegation***
A proxy credential does not have any binding to any actual Grid job or task. The availability or presence of a Grid user's proxy credential is no binding

statement assuring the authenticity of a Grid job or its sound processing.

Consequently, proxy credentials can be potentially stolen, misused or mixed up without notice at various points between the user's job submission and a WN. Similarly the description or payload of a job could be altered or exchanged. In [85] this concern was raised as the necessity to trust a VO to provide flawless correlations between proxy credentials and Grid jobs. Hence, the use of proxy credentials is not able to fulfill the security objectives of authenticity and non-repudiation of Grid jobs (see Section 4.5).

Vulnerability 6.2 ***Fuzzy validity and expiration*** The validity in time of a proxy credential is by itself independent of the validity or lifetime of a Grid job or any other submitted task.

Proxy credentials are propagated and stored at several services within a Grid layer, once a Grid user interacts with the system and submits e.g. Grid jobs. In the event of all Grid jobs or tasks of a user being terminated, corresponding proxy credentials within the Grid layer must be assumed to be still valid, at least in case of any intended storage of proxy credentials. The user has though no control over their deletion or invalidation and can generally not prevent their misuse.

Vulnerability 6.3 ***High dependence on VO and Sites and challenge of protection*** The access of a VO and Sites to Grid user credentials enables misuse by personal or attackers and the protection of proxy credentials during propagation or at rest becomes a critical concern.

Even if proxy credentials are never stored or processed within a VO's Central Grid services, as in case of the indirect user proxy propagation, a user or administrator holding certain privileges within the VO must still be considered as being able to obtain Grid users' proxy credentials, e.g. via access to necessary passwords. Moreover, a Grid job submitter's proxy credential will be accessible e.g. by the Grid Job Agent on a WN and consequently, everybody with according access to such system entities will be able to retrieve these proxy credentials. Thus, both system administrators as well as successful attackers of entities processing or storing proxy credentials are able to retrieve and deliberately misuse any stored proxy credential.

Vulnerability 6.4 ***Additional complexity, dependency and invocations*** The indirect user proxy propagation involves service invocations with a strong dependency of a Grid layer on the availability of the credential management service and the scalability of the mechanism. Also the potentially necessary renewal of proxy credentials introduces additional callbacks between Grid services.

Although these concerns might be considered a lower-ranking or circumstantial vulnerability, they impact not only availability and system performance as such, but as well affect the potential for errors and flaws within the infrastructure's implementation, operation and administration.

6.5 Summary

In this chapter, X.509 proxy certificates as a mechanism for delegation of privileges have been analyzed and their shortcomings and limitations have been specified.

Due to the conceptual limitations of unrestricted delegation, transparent user delegation based on X.509 proxy certificates leads to no significant improvement in comparison to a masquerade-based delegation approach. While the presence of a respective X.509 proxy certificate might be assumed to indicate or prove a Grid user's request to execute a certain Grid job or task within a Grid layer, there is no reasonable justification for such an assumption. Within a Grid layer utilizing delegation by masquerade, Grid job or task requests can potentially be altered or forged by at least administrators and successful system attackers (see Section 5.3). This is equally the case in a transparent user delegation scenario based on X.509 proxy certificates. Worse, the universality of an X.509 proxy certificate and its independent validity can potentially simplify and disguise an abuse or even enable an additional range of exploitation, in comparison to the possible misuse or alteration of a Grid job or task request within a delegation by masquerade. Finally, the utilization of X.509 proxy certificates implicates a raised scalability and availability dependency due to additional service invocations and callbacks.

X.509 proxy certificates based on unrestricted delegation are unprotected credentials, passed on and thereby exposed within a Grid layer. In consideration of the defined security objectives (see Section 4.5), X.509 proxy certificates represent insufficient assurance concerning authentication and authorization of any requests and are insignificant in matters of the authenticity and non-repudiation of Grid job submissions.

"Let every eye negotiate for itself
and trust no agent."
William Shakespeare

7 A Framework for Transparent Dynamic Delegation

In response to the deficiencies and vulnerabilites identified for both delegation by masquerade and delegation based on X.509 proxy certificates (see Chapter 5 and Chapter 6), a new framework for delegation in e-Science Grids will be presented in this Chapter.

After a discussion of related work, object entities within an e-Science Grid will be analyzed and distinguished. Based on these entities and the concept of signed textual statements, a new delegation framework, named *mediated definite delegation*, will be presented. The framework affords a transparent dynamic delegation of tasks via intermediary brokers and enables verifiable authenticity and non-repudiation of the authorship of Grid files as well as of submission and processing of Grid jobs.

Earlier versions of parts of this chapter have been presented in [PUB5, PUB10].

7.1 Related Work

Beyond the analysis of delegation by masquerade and delegation based on X.509 proxy certificates in the previous chapter, this section presents a discussion of further related work.

Aside from the defined security objectives (see Section 4.5), an auxiliary concern and boundary condition is the least invasive integration of security mechanisms into the defined e-Science Grid architecture (see Section 3.3). The utilization of remote callbacks or additional service invocations will induce additional complexity and risks of reduced availability due to potential failures or attacks, and will amount to additional dependencies and load in matters of scalability and delays.

In [44], a security framework for *on-demand restricted delegation* is presented as an approach in order to allow least-privileged delegation in Grid infrastructures. The mechanism is based on a delegation ontology, allowing to express the terms of delegations in a structured and formalized manner. Hence, it is for instance possible to specify *Subjects* as authorized delegatees or to explicitly express delegated *Capabilities* consisting of *Verbs* and *Objects* as authorized actions with respect to a delegation. Upon delegation of a task by a Grid user, the framework requires the user's client to provide a delegation service with a delegation policy for the respective task. Once services want to perform the task as a delegatee in the name of the user they are required to request a signed delegation credential from the delegation service, which validates the request based on the provided policy information. Despite the rich functionality and potential to provide context- and delegatee-sensitive least-privileged delegation, two fundamental drawbacks of the framework can be identified: first, the necessary translation of every delegation into a delegation policy, and second, the additional service invocations concerning the delegation service upon the publishing of a policy and the request of a delegation credential.

Within the UNICORE Grid middleware, a delegation framework called *explicit trust delegation (ETD)* [124] offers static delegation by digitally signing task requests using public-key signatures. ETD uses one signature, either by the Grid user or a trusted Grid portal, which in the latter case is consequently based on unrestricted delegation to the portal. Being based on a static delegation, ETD supports no intermediate processing of a task request.

In version 6 of the UNICORE framework, the delegation framework was revised, resulting in a new framework called *dynamic ETD* [52]. The new framework utilizes chains of signed assertions as operation requests based on public-key signatures and thereby provides for non-repudiation of the endorsement of an assertion. Yet, due to the necessary signature and verification on the level of single operation requests, the authors identify a drastic impact on performance. Moreover, the framework does not consider the authenticity and authorship of Grid files referred to within the assertions.

In [46, 45], a security framework for the Condor distributed batch computing system is presented, based on signed task descriptions, so called *Signed ClassAds*. A Signed ClassAd is placed inside an X.509 proxy certificate as a policy information, together with so called *action authorization expressions*. These are rules expressing which entity is allowed to use the proxy credential for which purpose. The mechanism enables e.g. to specify file checksums as conditions for executables and input files of a task. However, there is no explanation how to establish a dynamically assigned delegation or allow transformations of a Signed ClassAd. The framework's design seems to presume all conditions to be expressed explicitly upon the initial submission.

7.2 Object Entities of an e-Science Grid

Considering the classification of object entities within in a Grid layer, two fundamental classes of object entities can be distinguished with respect to their state and behavior while being processed within the system. These two classes are specified and defined as follows: any data set or program code within the Grid layer represents a *passive* entity *at rest*. Thereby any alteration of the content of these entities will change their identity, which implies any change of an existing object to result in a new object. This characterization conforms with the definition of data checksums.

Definition 7.1 Data objects like data and program code files and software packages are defined as *passive stationary entities*.

Grid Jobs represent *active* entities utilizing passive stationary entities, as non-trivial jobs are based on data and program code provided within the

Grid layer and Grid job results are stored in new data entries. Moreover, Grid jobs are *transient* as their benefit or utility is established only by their successful termination and the generation of corresponding results as data entries. Also Grid transfers represent such transient active entities as requests delegated to an agent to relocate or replicate physical copies of Grid file entries as passive stationary entities.

Definition 7.2 Processes or tasks like Grid jobs or Grid transfers operating on stationary entities are defined as *active transient entities*.

Further on, delegation within a Grid layer applies only to active transient entities directly while passive stationary entities are only affected as objects, as either being read, executed, modified or created. Whereas in essence a Grid user's interactions with a Grid layer can be characterized as the creation, modification and readout of passive stationary entities and the creation and delegation of active transient entities.

7.3 Framework of Mediated Definite Delegation

Within this section, a delegation framework named *mediated definite delegation* will be presented, providing transparent (see Section 6.1), dynamic and least-privileged delegation of active transient entities and assured authorship of passive stationary entities. The framework specifies the necessary steps and statements as a theoretical concept based on public-key signatures. With respect to the signature mechanism there are no additional assumptions, except to require each signor to secure its private key and the respective public key as a certificate to be available for verification of signatures wherever necessary within an infrastructure.

The defined e-Science Grid scenario (see Section 2.3) and architecture (see Section 3.3) is based on a three-tier architecture (see Section 4.2) of Grid users as service consumers, hereafter referred to as *delegators*, a central broker with processing services within a Virtual Organization (VO), hereafter referred to as *broker*, and Sites as resource providers with Worker Nodes (WNs), herein represented as *agents*. Further, Grid jobs or transfer requests as active transient entities are assumed to be represented as explicit textual specifications, which are modified within the different stages of processing and execution.

The framework of mediated definite delegation is based on the delegation of active transient entities by warrant, rather than employing auxiliary mechanisms, such as additional credentials or externally provided policies (see Section 4.1). In a nutshell, the outline of the framework is as follows: a delegator specifies an activity in a delegation request and sends it to a broker. The request is authenticated and authorized by the broker and consecutively processed. The processing may apply transformations to the request, while these transformations are required to be verifiable later on based on a predefined rule set. The broker subsequently selects an agent for execution and sends the transformed request to the agent. Thereby an implicit authorization of the request occurs, viz. the broker attests its approval by sending the request. The agent authenticates and authorizes the request and subsequently executes it. Any access to passive stationary entities during the execution of the request is assumed to be authenticated and authorized based on the delegation request as both a credential and a reference. Moreover, the agent issues a statement of authorship for any created passive stationary entity.

In the following paragraphs, the framework will be specified, beginning with the authorship of passive stationary entities, followed by a trivial introductionary case of a static delegation of active transient entities by warrant. Subsequently, the mediated definite delegation will be specified for one central broker as referred to above. Finally, the framework will be extended to a potential scenario of collaborating VOs with multiple brokers between delegators and agents.

Authorship. Within the framework, a statement of authorship is generally required for any creation of a passive stationary entity, e.g. file entries in the Grid File Catalog, and must be signed by a respective creator. With respect to passive stationary entities a declaration of authorship is defined as follows:

Definition 7.3 A *declaration of authorship* is defined as a creator-signed textual statement specifying the identity of the creator of the referred passive stationary entity as well as the name and identity of the passive stationary entity and the time of creation.

Every such statement then certifies the authorship of a specified passive stationary entity by a creator at a certain point in time. Figure 7.1 shows a Unified Modeling Language (UML) diagram of such a statement and its reference to a passive stationary entity.

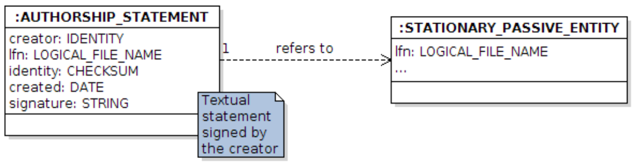


Figure 7.1: UML diagram of the declaration of authorship.

Requiring a digital signature of the statement by the creator, such a signed statement can prove the creator's certification of authorship later on as well as any potential changes of the entity. Within the limitations of a utilized digital signature mechanism the creator is thereby unable to repudiate the authorship.

Static delegation. Concerning the delegation of active transient entities, first, a trivial static delegation is defined and specified as a reference point and introductionary case:

Definition 7.4 A static delegation by warrant is defined as a delegator-signed textual statement specifying the delegator's identity, the delegated active transient entity as the job or task to be executed, the identity of the agent assigned for the execution, the time of creation and the time of expiration.

Every such statement then describes the delegation of a specified active transient entity by a delegator to an agent with a specified period of validity for its execution. Figure 7.2 shows a UML diagram of such a statement representing an active transient entity which itself may refer to passive stationary entities.

Requiring a digital signature of the statement by the delegator as the issuer, an agent receiving such a signed statement can prove its mandate

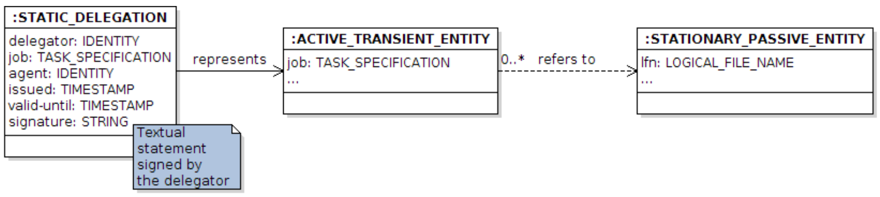


Figure 7.2: UML diagram of a static delegation.

later on and is able to use the signed statement as a credential. Within the limitations of a utilized digital signature mechanism the delegator is thereby unable to repudiate the submission and delegation of the specified task or active transient entity to the agent.

Dynamic delegation. In order to enable a dynamic delegation, allowing an intermediate broker to apply transformations to a delegator's initially submitted task or active transient entity and subsequently select an agent for its execution, a *mediated definite delegation* is defined in a two-step scheme as follows:

Definition 7.5 An *un-mediated delegation* is defined as a delegator-signed textual statement specifying the delegator's identity, the delegated active transient entity as the task to be executed, the identity of the broker that is assigned for the task's processing and later selection of an agent for execution, the time of creation and the time of expiration.

Every such statement then describes the delegation of a specified active transient entity by a delegator to a broker with a specified period of validity for its processing and later execution. Figure 7.3 shows a UML diagram of such a statement representing an active transient entity which itself may refer to passive stationary entities.

Requiring a digital signature of the statement by the delegator as the issuer of the statement, a broker receiving such a signed statement can prove its mandate later on and is able to use the signed statement as a credential for assigning the specified task to an agent. Within the limitations of a utilized digital signature mechanism the delegator is then unable to repudiate the submission and delegation of the specified active transient entity to the

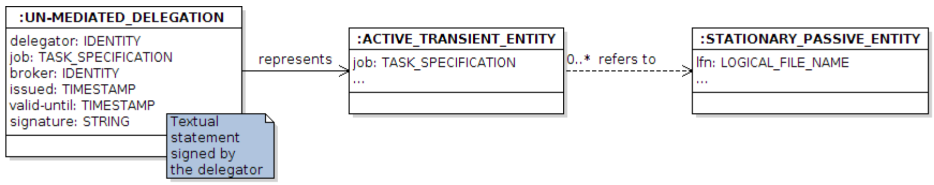


Figure 7.3: UML diagram the un-mediated delegation.

broker.

Definition 7.6 A *mediation of delegation* is defined as a broker-signed textual statement specifying the mediating broker's identity, the signed statement of the respective un-mediated delegation, the changes or transformations applied by the broker with respect to the initial task specification, the identity of the agent selected and assigned by the broker for execution, the time of creation and the time of expiration.

Every such statement then describes the mediation of a specified un-mediated delegation and the respective active transient entity by a broker to an agent with a certain period of validity for its execution. Figure 7.4 shows a UML diagram of such a statement which contains the un-mediated delegation it is descended from. The statement then represents an active transient entity which was changed by the transformations specified within the mediation of delegation.

Requiring a digital signature of the statement by the broker as the issuer and mediator, an agent receiving such a signed statement can prove its mandate later on and is able to use the signed statement as a credential during execution. Within the limitations of a utilized digital signature mechanism the broker is thereby unable to repudiate the mediation of the specified task or active transient entity to the agent. Equally, the delegator's non-repudiable initial submission and delegation of the specified active transient entity to the broker can be made transparently visible and verifiable for an agent, due to the cascaded signatures.

A dynamic transparent delegation within an e-Science Grid with one broker between users as delegators and agents can now be established utilizing the statements of *authorship*, *un-mediated delegation* and *mediation of delegation*.

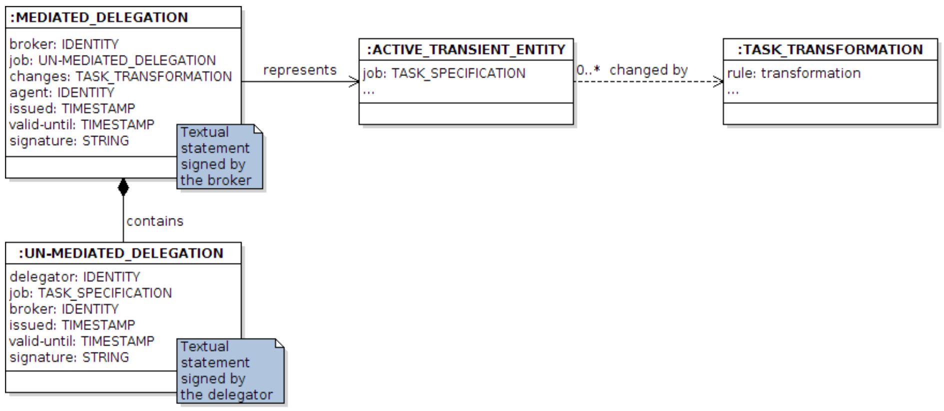


Figure 7.4: UML diagram of the mediation of delegation.

Multiple brokers. This delegation framework can be further extended to a scenario with multiple brokers, enabling transitive relationships between users as delegators and agents via more than one broker and their respective domains or VOs, as follows: due to the transitive relationship, a creator's identity might be unknown to a domain a passive stationary entity is used in. Therefore a witness of the authorship of a passive stationary entity is defined.

Definition 7.7 A *declaration of witness of authorship* is defined as a broker-signed textual statement, specifying the identity of the witnessing broker, the witnessed declaration of authorship, which can be a declaration of witness of authorship itself, and the time of creation.

Every such statement then certifies the witness of a specified declaration of authorship by a broker at a certain point in time. Due to the declaration of witness, passive stationary entities can be utilized beyond the borders of domains or VOs, with intermediate brokers vouching for their utilization. Figure 7.5 shows a UML diagram of such a statement which contains the declaration of authorship it is descended from. The declaration of witness of authorship is further declared to be a subclass of the declaration of authorship. Therefore, it is possible to concatenate declarations of witness of authorship via more than one broker.

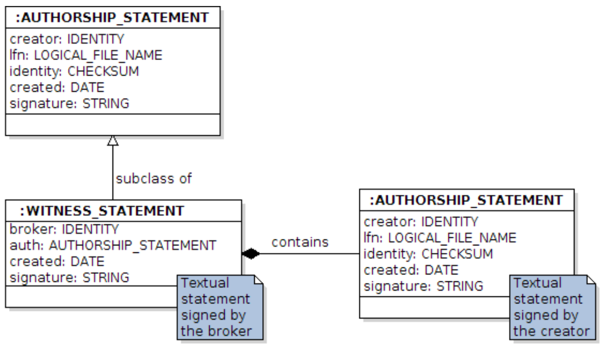


Figure 7.5: UML diagram of the declaration of witness of authorship.

Requiring a digital signature of the statement by the broker, such a signed statement can prove the broker's certification of a witness of authorship later on as well as any potential changes to the passive stationary entity concerned. Within the limitations of a utilized digital signature mechanism the broker is thereby unable to repudiate the witnessing.

In order to further allow for the transitive relationship of users as delegators and agents via more than one broker, a propagation of un-mediated delegation between brokers is defined:

Definition 7.8 An *un-mediated propagation* is defined as a broker-signed textual statement, specifying the propagating broker's identity, a signed statement of an un-mediated delegation, the changes or transformations applied by the broker with respect to the foregoing task specification, a second broker's identity to which the statement is propagated to, and the time of creation as well as the time of expiration. The signed statement of an un-mediated delegation can thereby be an un-mediated propagation itself.

Every such statement then describes the propagation of a specified un-mediated delegation and the respective active transient entity by a broker to another broker with a specified period of validity for its processing. Figure 7.6 shows a UML diagram of such a statement which contains the un-mediated delegation it is descended from. The un-mediated propagation is further declared to be a subclass of the un-mediated delegation. Hence, it is possible to allow more than one propagation or respectively a concatenation of propagations via brokers up to a final mediation of delegation.

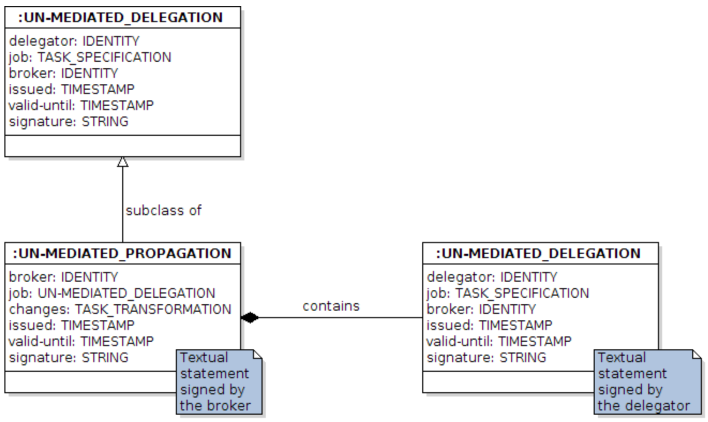


Figure 7.6: UML diagram of the un-mediated propagation.

Requiring a digital signature of the statement by the broker as the issuer and propagator, a second broker receiving such a signed statement can prove its mandate later on and is able to use the signed statement as a credential during mediation or further propagation. Within the limitations of a utilized digital signature mechanism the first broker is thereby unable to repudiate the propagation of the active transient entity to the second broker. Equally, the delegator's non-repudiable initial submission and delegation of the specified active transient entity to the broker and potential consecutive propagations by brokers can be made transparently visible and verifiable to an agent.

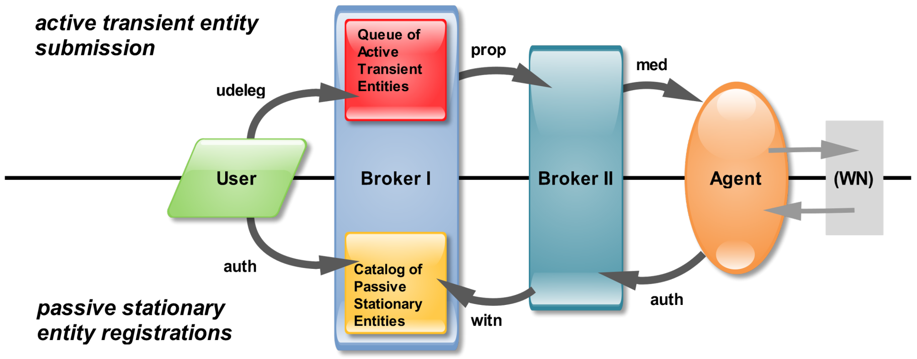

active transient entity submission
udeleg
Queue of Active Transient Entities
prop
med
User
Broker I
Broker II
Agent
(WN)
auth
Catalog of Passive Stationary Entities
witn
auth
passive stationary entity registrations

Figure 7.7: The framework of mediated definite delegation with two brokers.

Figure 7.7 illustrates the concept of the framework of mediated definite delegation as a whole with the defined signed statements passed on in interaction requests. The submission, processing and assignment of an active transient entity by a user is demonstrated by the request *'udeleg'* as an un-mediated delegation to a queue of active transient entities, such as a Grid Task Queue, at the first broker. The first broker initiates a propagation of un-mediated delegation as the *'prob'* request to the second broker. The second broker finally arranges the mediation of delegation by the *'med'* request to the agent, where the active transient entity can be executed, e.g. as a Grid job on a Worker Node (WN).

The declaration of authorship of a passive stationary entity, e.g. a file entry, by a user is shown as an *'auth'* request to the first broker managing a catalog of passive stationary entities, such as a Grid File Catalog. The declaration of authorship for an active transient entity result, e.g. as an output file, is represented by the agent's *'auth'* request to the second broker. The second broker then creates a declaration of witness of authorship represented by the *'witn'* request to the first broker.

The figure shows an active transient entity queue and a passive stationary entity catalog within the first broker. With according adjustments of the respective *'witn'* and *'prob'* requests, one or both of them can be alternatively located at any other or further brokers.

7.4 Summary

A new framework for delegation in e-Science Grids named *mediated definite delegation* has been presented in this Chapter. The framework allows to transparently delegate active transient entities, viz. tasks or Grid jobs, via one or more brokers to an agent for execution. Any intermediary broker can perform transformations of the initial specification of an active transient entity and dynamically select either another broker for further propagation of the request or an agent for execution. Based only on signed statements which are sent out by a statement's signer, viz. a user or broker, a receiver of such a statement, viz. a broker or agent, is able to verify the digital signature and all relevant information provided within the statement. Hence, the authenticity of requests and according authorizations can be immediately verified and non-repudiation of the requested actions is provided.

For passive stationary entities, such as Grid files or software packages, the framework requires a creator-signed declaration of authorship affording non-repudiation of the creation and verifiable authenticity of a passive stationary entity. Once passive stationary entities are utilized within domains of no direct relation to the entities' creators, declarations of witness of authorship that are signed by witnessing intermediaries can be used. Hence, transitive assurance via several intermediary witnesses of the authorship can be arranged.

Due to the presented framework, the receiver (agent) of a statement of mediated delegation can be forced to use the signed statement as a credential to access passive stationary entities during the execution of the respective active transient entity, for instance with respect to file or data access. Thereby, any authorization is bound and limited according to the specifications of the statement and an agent is consequently not authorized to execute arbitrary program code or make arbitrary file or data changes. Accordingly, the statement as a credential affords protection against misuse of the delegating user's identity as well as against unnoticed alteration of the requested actions. Yet, this requires the specifications of a statement being sufficiently detailed and nonambiguous to facilitate an according least-privileged delegation.

The framework and its mechanisms completely integrate into the e-Science Grid architecture defined in Chapter 3 and require no remote callbacks or additional invocations.

"Love all, trust a few, do wrong to none."
William Shakespeare

8 A Security Architecture for e-Science Grids

In this chapter a new security architecture for e-Science Grids established on the framework of *mediated definite delegation* (see Chapter 7) will be presented and specified in detail.

The delegation framework as a concept will be applied to the defined e-Science Grid architecture (see Section 3.3). The resulting security architecture for e-Science Grids will extend mechanisms implemented within the ALICE Grid Services (see Section 3.1) and include solutions to the identified vulnerabilities (see Chapter 5). Beyond the framework of mediated definite delegation, the security architecture for e-Science Grids will thereby provide the fulfillment of the defined security objectives (see Section 4.5).

The chapter is organized in three main sections concerning Grid data layer security, Grid job layer security and secured communication and will conclude in a discussion of the results and achievement of the presented security architecture.

Earlier versions of parts of this chapter have been presented in [PUB4, PUB5, PUB7, PUB9, PUB10].

8.1 Grid Data Layer Security

Within the following sections, the framework of mediated definite delegation (see Section 7.3) will be applied to the defined e-Science Grid architecture's data layer (see Section 3.3). The presented mechanisms refer to the Grid File Catalog and the SE access protocol of the ALICE Grid Services (see Section 3.1.3) and include solutions to the identified vulnerabilities (see Section 5.2). Further, multiple cascaded Grid File Catalogs will be discussed as an extension.

First, a mechanism for Grid file system consistency in reply to the identified vulnerabilities in the ALICE Grid Services will be presented. Thereafter, an implementation of the authorship statement for passive stationary entities within the framework of mediated definite delegation will be specified, which introduces certified Grid files. Subsequently, a mechanism for on-demand storage discovery will be presented, affording improvements with respect to storage quality-of-service and availability. Further, an implementation of Grid file transfers will be briefly specified.

8.1.1 Grid File System Consistency

Due to the separation of a Grid file system into independent layers of logical and physical data based on a Grid File Catalog and SEs (see Section 3.3), a two- or three-phased protocol for data access is required: for read access, a client or service retrieves logical information and authorization on the level of Central Grid Services based on the Grid File Catalog and consecutively connects to an SE and reads the data. In case of a write access, the client or service requests authorization on the central level, consecutively connects to an SE and writes the file entry, and finally registers a new file entry or respective modifications on the central level in the Grid File Catalog.

Within this schema, race conditions or collisions concerning logical Grid file entries can occur during write operations, which was identified before as a vulnerability within the ALICE Grid Services (see Section 5.2). A similar functional issue affects the deletion of logical and physical file entries and the potential reuse of previously existing Logical File Names (LFNs) in the Grid File Catalog.

Booking Table An LFN entry can refer to multiple Physical File Names

(PFNs) as replicas of the same physical data (see Section 3.3). Once a client requests access to a new file entry (LFN) in the Grid File Catalog, the LFN will be in a pending or transient state conceptually until the client uploaded the physical file and triggers its registration. The same applies to the respective PFN entry, as a replica of the LFN. In order to keep track of write or delete operations in a Grid file system, a database table, named *Booking Table*, is introduced. Whenever a PFN entry enters or leaves the Grid File Catalog, entries in this table present the PFN in its transient state until the final completion of the operation. The state of an LFN is implicitly represented by its PFN entries, once it is not existing in the Grid File Catalog.

Figure 8.1 shows the state modeling of LFN entries and their transitions, which is explained as follows: Upon an authorized write request from a service or client, a corresponding entry is added to the Booking Table for each PFN, containing the PFN, LFN and owner of the prospective Grid File Catalog entry. This alters the LFN entry's state from *untaken* to *booked* in case it is unknown to the system, or from *freed* to *booked* in case it was used before or deleted (see below). Upon registration as the finalization of a write operation, an entry is removed from the Booking Table and registered in the Grid File Catalog. This alters the LFN entry's state from *booked* to *visible* once one PFN entry is registered.

In case a user removes a file entry or a PFN entry as one replica of an LFN entry in the Grid File Catalog, the physical removal is independent of the Grid File Catalog. Again, an entry for each concerned PFN is added to the Booking Table, and the LFN is removed from the Grid File Catalog, once the last PFN was removed. The latter is represented by the state transition from *visible* to *freed*.

The Booking Table further has a lifetime value for each entry whereby a time out for unregistered write access requests can be provided, which describes the distinction of the LFN states *booked* and *freed*. Potentially existing unregistered data entries on SEs can then be removed based on the Booking Table entries, represented in the LFN state transition *freed* to *untaken*. Alternatively, in case a *freed* LFN is requested by a user (again), the LFN will be set to state *booked*.

By introducing the Booking Table, LFNs and PFNs can be represented in all their respective states, file operations altering the Grid File Catalog can be assured to be atomic and their access control can be consistently enforced.

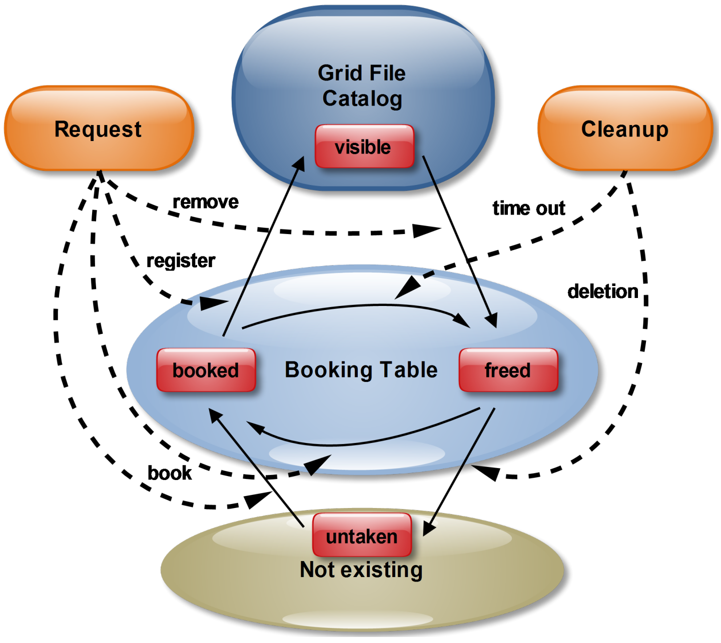


Figure 8.1: The states and transitions of an LFN entry in the Booking Table.

8.1.2 Certified Grid Files

Once a file operation is authorized on the level of Central Grid Services and the Grid File Catalog, a client has to be enabled to access an SE and perform the physical file operation. In order to avoid additional callbacks or invocations in terms of authentication and authorization an *Access Ticket* is defined, initially based on the concept of an Access Envelope [74] as utilized within the ALICE Grid Services (see Section 5.2).

Access Ticket An Access Ticket grants direct access to a specified file entry on an SE, according to its internal authorization information. It specifies the PFN as the location of the physical file entry, the granted access permission and timestamps of creation and expiration. Based on a public-key signature mechanism, the ticket is signed after authorization by a Central Grid Service, using the private key corresponding to an X.509 certificate of the VO's Central Grid Services. Once sent to a client upon request, the signed Access Ticket will allow a client to authenticate and authorize an according access at an SE.

Status Ticket Due to the decoupling of logical and physical data, Central Grid Services rely on client-provided information concerning identity and size of a file. In order to provide for a confirmation of this information by an SE, a second SE-signed ticket is utilized. Thereby each SE uses its own private key, with the corresponding public key being available to Central Grid Services, e.g. based on an X.509 certificate. Within this *Status Ticket*, the SE states the PFN of a file entry and the verified checksum and size of the according file data, together with the timestamp of creation of the ticket. After a successful write operation of a file entry on an SE by a client, the client requests the signed Status Ticket from the SE and sends it to the Central Grid Services for registration of the file entry in the Grid File Catalog. Thereby the Status Ticket assures the existence, size, and checksum of the respective file entry.

File Certificate The usage of Booking Table, Access Ticket and Status Ticket in combination can assure that a Grid File Catalog contains only consistent and authentic data with respect to the physical storage level of SEs. In order to further allow for undeniable authorship of Grid file entries, the final registration of a new or changed file entry in the Grid File Catalog additionally requires a signed request to be sent by the Grid Client Interface to Central Grid Services, named *File Certificate*. A File Certificate contains a respective file entry's LFN, the data checksum, the username and a timestamp of creation, and is signed by Grid users using the private key corresponding to their X.509 Grid certificates.

Write Figure 8.2 shows the final access protocol based on Access and Status Ticket and a File Certificate in case of a write operation on a Grid file, which is further explained as follows: initially, the Grid Client Interface requests an Access Ticket from Central Grid Services. After successful authentication and authorization of the request, an according entry is added to the Booking Table. A signed Access Ticket is sent by Central Grid Services to the Grid Client Interface. The Grid Client Interface uses the Access Ticket to request write access at the SE, which grants the access based on the Access Tickets signed information. In case a physical file entry already exists, the SE consults the time of creation of both the physical file entry and the Access

Ticket. The access will then only be authorized in case the existing file entry was created before the Access Ticket. Thus, malicious replays of Access Tickets can be prevented.

Upon a successful write operation, the Grid Client Interface requests the Status Ticket from the SE, which calculates size and checksum of the file and creates and replies a signed Status Ticket. The Grid Client Interface assures the file operation was successful using the Status Ticket and creates a signed File Certificate, which is then sent together with the signed Status Ticket to the Central Grid Services for registration. If the verification of the Status Ticket and the File Certificate are successful, the corresponding entry in the Booking Table is deleted and the file entry is registered in the Grid File Catalog together with the File Certificate.

Delete The deletion of file entries is analogues to a write operation,

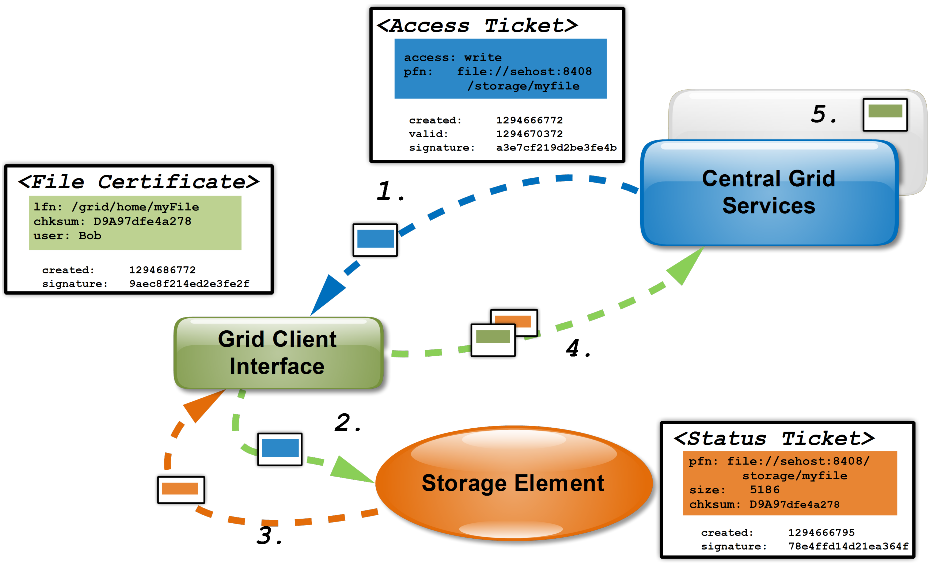


Figure 8.2: The certified Grid file write access based on a File Certificate.

while the SE sends a Status Ticket stating the physical deletion to the Grid Client Interface. The Grid Client Interface then sends the Status Ticket to the Central Grid Services, where the PFN is finally marked as deleted within the Booking Table (see Section 8.1.1).

Read Concerning read accesses of file entries, a Grid Client Interface requests an according Access Ticket from the Central Grid Services. In case of successful authentication and authorization of a request the Grid Client Interface uses the replied Access Ticket to request the physical file data from an SE.

Copying, moving and alteration Whenever a file entry is (physically) copied including a new LFN entry in the Grid File Catalog, viz. the file is neither replicated as a new physical copy of the same logical file, nor logically linked to an LFN, the according request and creation procedure is analogous to the creation of a new file. Consequently, the File Certificate of the new file entry will be signed by and represent the identity of the copying user. Analogously this applies to file entries that are logically moved within the file system hierarchy of the Grid File Catalog or to file entries whose data was physically altered or modified. There, the File Certificate will be signed by and represent the identity of the modifying user.

File Certificate storage The File Certificates are stored within the Grid File Catalog as an additional attribute of the respective LFN entry. Furthermore, the X.509 user certificates are presumed to be stored in a database or storage within the Central Grid Services.

Uncontrolled data locations The registration of uncontrolled or external data locations as PFNs in the Grid File Catalog, as e.g. data accessible via HTTP, is an additional concern, as it is neither possible to assure file data integrity with respect to a respective PFN entry, nor to identify its provenance. Any LFN registration of data entries not accessible via the previously defined access protocol based on trusted SEs does therefore require an additional verification: a registration of an uncontrolled data resource will require a Grid user-signed File Certificate stating the respective LFN and PFN, the data checksum, the username and a timestamp. Upon request an according entry is placed into the Booking Table, while a central verification service retrieves the file data from the resource specified by the PFN. The data checksum is verified using the provided File Certificate, and in case of successful verification the entry is removed from the Booking Table and added to the Grid File Catalog. In this case, the Grid user vouches for the registered external data resource and data integrity can be verified later on.

Cascaded Grid File Catalogs Concerning cascaded Grid File Catalogs (see Section 3.3), wherein file entries in global Grid File Catalogs refer to local Grid File Catalogs in terms of physical locations of file data, the usage of SE-based local Grid File Catalogs is assumed to be consistent with the previously presented scenario. Concerning global file entries, File Certificates must be equally provided by registration, in order to assure authenticity and authorship within the global Grid file system layer. This case is assumed to be either analogous to the registration of external data locations, with a central verification service retrieving and verifying the entry, or size and checksum information of the data being fetched directly from the according local Grid File Catalog, in case a respective trust relationship allows that.

In conclusion the propagation of a File Certificate represents a declaration of authorship (see Definition 7.3) within the framework of mediated definite delegation (see Section 7.3) with the respective File Certificate as the creator-signed statement. With the Status Ticket, digitally signed by an SE, a witness of authorship is further implemented similar to the declaration of witness of authorship (see Definition 7.7).

8.1.3 On-demand Storage Discovery

In order to enable transparent utilizations of physical file replicas, failover functionality and improved performance, a dynamic on-demand SE discovery mechanism is integrated into the SE access protocol.

Ranking By introducing SE status and ranking tables within central Grid services combined with scheduled availability and performance checkups, it is possible to select SEs for clients' data read and write requests based on multiple criteria: a main central status table contains information on availability, and bandwidth and usage statistics of all known SEs. This information is interlaced with network topology and performance statistics, such as geographic and routing proximity and round-trip times, in between all Sites within the Grid layer via the MonALISA framework [25, 3]. The results are stored relative to every Site within an SE ranking table, representing a preference list of all functional SEs for every Site. Further, SEs can be labelled with storage identifiers, such as disk or tape storage type, as well as access restrictions via their registration record in the Grid Layer's central LDAP directory.

Discovery Upon read or write requests by clients, whether Grid users or job clients, the respective client IP address is mapped to a Site based on network topology. During the processing of the request, prior to the creation of an Access Ticket, an SE is selected based on the SE ranking table, using the mapped Site as the key attribute. According to the type of request, the best SE entries are selected for the consecutive creation of Access Tickets. Thereby, access restrictions for SEs and possibly also additional parameters specified by the client or centrally predefined rule sets are taken into account. Parameters specified by a client within a request can be explicitly favored or excluded SEs, SEs that failed during previous requests as well as storage types of SEs according to the predefined labels. In case of a write request by a client, a number of intended replicas of a file can be specified. Centrally predefined rule sets can concern e.g. load-balancing or data distribution as further emphasized below.

In case of a read request of a client for Grid file entry, with several replicas of the physical file on different SEs, the mechanism is accordingly able to dynamically select the best available SE with respect to the client. In case of a write request for a new file entry, the mechanism chooses the best available SEs and consecutively issues and replies all respective Access Tickets within one request. In case of failing file operations on SEs, a client simply requests again Access Tickets for SEs, specifying the respective SEs' identifiers as pre-failing SEs within its consecutive request for Access Tickets. Thereby, the Booking Table (see Section 8.1.1) assures consecutive write requests for the same logical file are consistently handled with respect to the File Catalog. Hence, once a request to an SE fails, the mechanism enables a failover to another SE whenever possible, and thereby avoids potentially failing client operations due to unreachable data or impossible data saving.

Data distribution Further on, an automatic data distribution can be achieved, leading to improved data availability as both data access performance and prevention of data loss: the presented mechanisms allow to automatically store Grid job results close to the WN, at which a respective job is executed by the selection of network-topology-wise close SEs. Hence, with an according Grid job management it is possible to arrange the results of Grid jobs to be stored in proximity of the respective Grid jobs' input files. This approach can then be capitalized in case of subject- or content-wise-clustered data files or a multistage processing of data. At the same time, the mechanism allows to implement further constraints for data distribution via predefined rule sets respected during the selection of SEs. Examples of such constraints are storage types, e.g. a tape storage, or regional distance with respect to the client, such as storage on another continent, geographic region or country.

8.1.4 Grid File Transfers

Grid file transfers are classified as active transient entities (see Section 7.2) and could be implemented analogous to the framework's implementation of Grid jobs specified below in Section 8.2. The functionality of Grid file transfers (see Section 3.3) in combination with the SE access protocol specified above allows though for a conceptual simplification, since Grid file transfers only replicate or move physical data of existing file entries in the Grid File Catalog from one SE to another.

Due to the presented mechanism of certified Grid files, a file entry's authorship and the authenticity of respective physical file data are verifiable, impeding covert alteration of logical file entries and physical file data as well as forgery of logical file entries. Hence, an implementation of Grid file transfers is only required to include the verification of the SE-signed Status Ticket, as specified above. In this case, a Grid file transfer delegated to any agent as a replication or moving of physical file data is strictly limited and controlled, including the event of an unsuccessful operation.

8.1.5 Implementation in AliEn

In AliEn version 2.19, the Booking Table and a first stage of the certified file operations as well as the on-demand storage discovery were implemented and deployed to the ALICE Grid Services in production [PUB4, PUB1]. The Access and the Status Ticket where fully implemented, while the implementation did not include the File Certificates. The Status Ticket registration was not enforced, and the mechanisms were applied in order to merely assure consistency of the logical information in the Grid File Catalog and physical file entries on SEs.

8.2 Grid Job Layer Security

Within the following sections, the framework of mediated definite delegation (see Section 7.3) will be applied to the defined e-Science Grid architecture's job layer (see Section 3.3). The presented mechanisms include solutions to the vulnerabilities identified within the ALICE Grid Services (see Section 5.3).

First, an implementation of the dynamic delegation of active transient entities within the framework of mediated definite delegation will be specified, leading to the concept of certified Grid jobs. This will be followed by a brief consideration of Grid applications security. Finally, two approaches for Grid job execution protection will be presented and outlined.

8.2.1 Certified Grid Jobs

Analogous to the concept of the File Certificates presented beforehand, Grid Client Interface can digitally sign a JDL upon submission of a Grid job request. By demanding a user-signed JDL (hereafter referred to as an *sJDL*) for the submission of a Grid job, it is possible to prove the integrity of a Grid job and to identify illegal modifications with respect to the user's submission at any point in time later on. Using timestamps within the sJDL, set by the Grid Client Interface and verified by a VO's Central Grid Services upon submission, an sJDL represents a delegation by warrant, stating a user requested a certain Grid job to be executed within the specified time frame.

Job transformations According to the defined e-Science Grid architecture (see Section 3.3), a Grid job submission and respectively its JDL can be transformed during its processing within Central Grid Services. Such transformations are applied to an sJDL by appending changed or additional statements, which will overrule the initial specifications of the sJDL of the submitter. In order to allow for later verification of such transformations' correctness, all transformations must comply with a predefined rule set. An example of a transformation is the splitting of a job into sub jobs according to the distribution of input data: the original sJDL is thereby copied for every sub job, and a determined input data specification is appended to each sub job's sJDL, which overrules existing input data specifications of the original part of the sJDL.

Job Certificate Before a Grid Job is sent to a Site for execution, a VO's Central Grid Services (as a broker) digitally sign a processed and transformed sJDL, using the private key corresponding to an X.509 certificate of the VO's Central Grid Services. This double-signed JDL will be hereinafter referred to as a *Job Certificate*.

Once arriving on a Worker Node (WN), an authenticated Grid Job Agent (GJA) gets authorized as a delegatee of a Grid job, which the GJA is requested to execute according to statements defined within the Job Certificate.

Job submission A protocol for certified Grid Jobs based on Job Certificates is illustrated in Figure 8.3 and presented as follows: upon submission of a Grid job, the Grid Client Interface sets a submission and an expiration timestamp in the JDL and signs it using the Grid user's private key, corresponding to its X.509 certificate. This sJDL is sent to the Central Grid Services as a Grid job submission. If the Grid user's X.509 certificate is not in the VO's disposal already, the X.509 certificate is appended to the submission request. The signature of the sJDL and the submission and expiration timestamps are validated within the Central Grid Services, the sJDL is placed into the Grid Task Queue and subsequently processed.

Job execution As a result of a previous request from Central Grid Services, a Site's Grid Site Service submits a Grid Job Agent (GJA) request to the Site's resource management system. The GJA request includes the Central Grid Services' X.509 certificate and a credential in order to allow the later GJA to connect to Central Grid Services. Upon startup, the GJA connects to Central Grid Services, and authenticates itself using the credential and requests a Grid job for execution.

Presuming a successful matchmaking between the Grid job request's requirements and the GJA's capabilities, the processed (and thereby possibly transformed) sJDL is prepared for submission to a Site: after appending anew submission and expiration timestamps as well as an identifier of the Site (or the WN) to the sJDL, the Central Grid Services digitally sign the

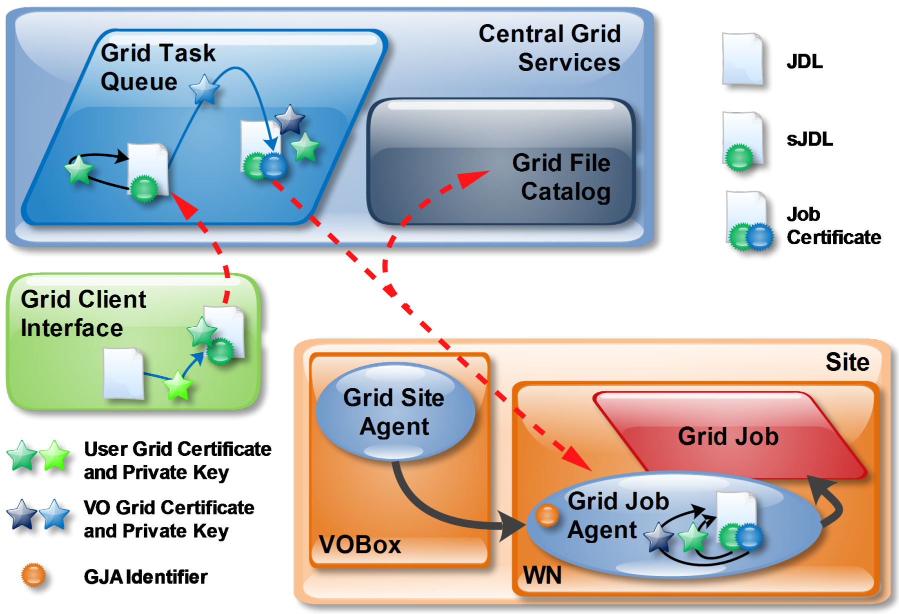


Figure 8.3: The processing of a certified Grid job based on a Job Certificate.

processed sJDL using the private key corresponding to a VO-specific X.509 certificate, which thereby becomes a Job Certificate. While the Grid user's submission and expiration timestamps within the Job Certificate define the time window the VO is entitled to send the Grid job request to Sites, the VO's timestamps can be adapted to the desired runtime window of a Grid job on a WN, and as such can define a shorter time window. Once created, the Job Certificate is sent together with the Grid user's X.509 certificate to the GJA. The GJA verifies the Job Certificate's signatures, the two time frames of validity and the X.509 certificates, and refuses to execute the Grid job in case of a failure in any of these validation steps. During the Grid job's execution, the Job Certificate is then used by the GJA as a credential in order to authenticate and authorize the access of Grid files, such as the job's input files. These authorizations occur based on the data specifications within the Job Certificate.

Job output files Once a Grid job has finished or stopped, the GJA uploads and registers the job's output files in the name of the Grid job submitter, using the Job Certificate as a credential. Thereby, job output specifications in the Job Certificate can assure a GJA will only be granted write access to the specified files or directories. Due to the concept of certified Grid files, also the file registration of output files requires the creation of signed File Certificates (see Section 8.1.2). These signatures can be applied either via a signature service controlled by the Site (see Section 8.4) or directly by the GJA, while an according X.509 certificate and key must be provided. Consequently, the certificate will belong to and represent a respective Site's or VO's identity at a Site. In order to identify the file entry's origin, the file entry in the Grid File Catalog must be further linked to the Job Certificate via meta data.

Job certificate logging In order to enable traceability and non-repudiation of Grid job submissions, the Job Certificate of an executed Grid job must be logged and kept by a Site. This will require a Site-provided facility to log a Job Certificate upon Grid Job execution, which will be further discussed in Section 8.2.3.

In conclusion, the presented mechanism using composite signatures of a JDL represents the instantiation of the framework of mediated definite delegation (see Section 7.3) with one broker between users and agents. Thereby, the user-signed sJDL represents a request of un-mediated delegation (see Definition 7.5), and the submission of a Job Certificate to a Site represents a mediation of delegation (see Definition 7.6). Within the framework, transformations of an initially delegated active transient entity are specified within the statement of mediation of delegation. This aspect is arranged slightly different within Job Certificates, as the results of transformations are appended to an initially signed job specification (sJDL), rather than providing the transformations as statements themselves.

8.2.2 Grid Application Security

During their storage as software packages, Grid applications represent passive stationary entities (see Section 7.2), and accordingly their integrity and authenticity is required to be protected (see Objective 4.2). In the case of Grid applications are provided via Grid file entries in a Grid File Catalog, verifiable integrity and authenticity is achieved by the presented mechanism of certified Grid files. Otherwise, a mechanism analogue to the File Certificate can be implemented, wherein a Grid application package's data checksum together with a timestamp is digitally signed using a known and acknowledged X.509 certificate upon deployment. The signature can then be verified before the application's use, e.g. by the GJA. Hence, a package's authenticity can be verified and alterations of Grid applications referred to by Grid jobs can be identified before their execution.

8.2.3 Grid Job Execution Protection

A remaining issue is the Grid job execution protection on a WN as the protection of the GJA against access from Grid jobs and the mutual isolation of Grid jobs (see Vulnerability 5.13). Such a protection cannot be afforded by the presented mechanisms and requires additional functionality to be emplaced, the utilization of according mechanisms is though foreseen and arranged. Moreover, with respect to traceability and non-repudiation of a job submission, a Job Certificate is required to be logged on Site-level, which is assumed to be handled by an according Site-based mechanism. As a reply to these concerns, two potential approaches are outlined within the following paragraphs.

gLExec the middleware component *gLExec* [86] facilitates the protection of a GJA and the isolation of Grid jobs on the level of an UNIX operating system's user account separation, similar to the UNIX `sudo` command. Instead of a direct execution of a Grid job on a WN, gLExec is invoked by a GJA, which performs a user and environment switch on a WN's operating system, and consecutively executes the Grid job as another system user. Moreover, gLExec can be configured to authenticate and authorize a job request, expecting an X.509 proxy credential as an additional input parameter, e.g. the proxy credential of the Grid job submitter (see Section 6.3). By appropriate modifications within gLExec, this authentication and authorization can be provided, based on a verification of Job Certificates. Moreover, a Site-level logging of Job Certificates could be directly integrated. Such a modified implementation of gLExec would then enable a Site to enforce access control with respect to job executions, while requiring no longer to entrust this control to the GJA service.

Virtualization Alternatively, Grid frameworks based on virtualization, like the CernVM Co-Pilot [59, 88, 87] and XGE [123, 122], facilitate management and submission of virtual machine images to WNs and the execution of Grid jobs within virtual machines. Thereby strong isolation of Grid jobs and consequent protection of a GJA can be achieved. The combination of the herein presented mechanisms and a Grid job execution based on virtualization is therefore assumed to make further security improvements possible. Whereupon, analogous to a solution based on gLExec, an implementation could include the Job Certificate as a credential for authentication and authorization of a virtualized Grid job execution and a logging of the Job Certificate.

8.3 Secured Communication

Mutually authenticated and protected communication of Grid services can be established with standard mechanisms like the *Transport Layer Security (TLS)* [70] protocol, while using X.509 certificates for authentication. Due to the availability of X.509 certificates on part of Grid users and the required X.509 certificate for Central Grid Services in terms of the creation of Job Certificates, secured communication can be established for communications between Grid Client Interfaces and Central Grid Services. Provided that X.509 certificates are also available for Site-based services, these communications can be equally protected, including the data communication with SEs. Concerning the LDAP directory service, a standard method [37] for authenticated and protected communication is available, utilizing the TLS protocol with X.509 certificates for authentication.

8.4 Discussion and Evaluation

The results and achievements of the presented security architecture established on the framework of mediated definite delegation with respect to the defined security objectives (see Section 4.5) are discussed as follows:

Certified Grid files Due to the presented mechanism for certified Grid files, a Grid File Catalog can be fully based on authentic Grid file entries, while the introduced Booking Table assures consistency and non-bypassable authorization. On the basis of Access Ticket, Status Ticket and File Certificate, it is possible to assure the authenticity of data on the logical level, while preserving the original two- and three-phased protocol, viz. requiring no additional callbacks or direct connections between the logical and physical data level. File Certificates in combination with trusted data checksum verifications force Grid users to acknowledge and attest the authorship of introduced file data. Hence, the mechanism provides verifiable integrity, authenticity and non-repudiable authorship of Grid files (see Objective 4.1).

Provenance of scientific data Scientific provenance frameworks [67, 80] provide mechanisms in order to retrace all steps of processing that produced a certain data set, and thereby facilitate reproducible findings. The presented security architecture can assure the origin of data concerning their authorship, or their origin by means of a certain Grid job execution. Consequently, it provides a universal basis for further analysis of scientific data provenance on the level of Grid files, independently of an area of research and its computational methods. A limitation concerns the copying, modification, deletion or moving of file entries and data, which will result in a consequently different identity of authorship, whenever an operator in place is not the author of the original file entry and data. Yet, this limitation can be overcome by according constraints concerning the authorization of such operations and procedures to retain original file data including the corresponding File Certificate.

Certified Grid jobs The introduced functionality of certified Grid jobs introduces no additional remote invocations or callbacks between Grid Client Interfaces and Grid Services and preserves the natural calling scheme of a job submission. Also, no renewal or refreshment or storage of exploitable user credentials is required, as the concept of certified Grid jobs as credentials for delegation grant no other privileges than the execution of respectively specified jobs.

The mechanism enables dynamic delegation and splitting of Grid job requests into sub jobs and affords traceability and non-repudiation of both submissions and intermediate processing (see Objective 4.4). Concerning the revocation of Grid users' access or corresponding X.509 certificates, previously created File and Job Certificates are not invalidated by a revocation and their authenticity and authorship persists.

Furthermore, due to the introduced functionality the necessary privileges of the GJA can be reduced according to the least-privilege principle, whereby a GJA requires no higher privileges than allowing the retrieval of Job Certificates as job requests. Hence, threats like the proliferation of attacks, e.g. due to anew Grid job submission, can be dissolved.

Based on the combination of File and Job Certificates, it is possible to assure not only a Job Certificate's authenticity and authorship, but also the authenticity and authorship of all Grid files referred to from within a Job Certificate. Therefore, it is possible to largely simplify the detection of illegal or improper behavior of the environment of a Grid job on a WN, being able to refer to a Job Certificate as well as File Certificates of involved file entries as the expected and requested behavior.

The significance of the mechanisms concerning non-repudiation and potential responsibility of Grid users in the face of security incidents, is discussed as follows: in case a hostile or malicious Grid job is executed, while presuming an environment of integrity of the Grid job on a WN, malicious code can be assumed to be either provided directly within the Grid files referred to via the specification of the job, or by reference to external data source, e.g. via the Internet. In the course of forensic analysis of the Grid job, evidence would need to be identified within the Grid job specification and its input files, which would be assumed to reveal either respective instructions or at least a request to retrieve data from external sources. In both cases, the Job Certificate and File Certificates together facilitate non-repudiation of the respective submission. In case a Grid job refers to or involves malicious data or operations not connected to the job submitter, the responsibilities can be clearly identified. Also the transformations of an original submission can be transparently identified and verified later on based on the cascaded signatures in the Job Certificate.

A remaining issue concerns the possible deletion or overwriting of Grid files, e.g. as a measure in order to cover up traces. Although, based on the runtime of a Grid job and the modification timestamp of Grid files in the File Certificate, such an event could be possibly identified, the original or foregoing content of a deleted or modified Grid file could presumably not be recovered. Potential solutions could be a temporary blocking of deletion requests or a deferred deletion functionality concerning Grid files 'recently' utilized within Grid jobs.

Finally, in the events of e.g. malicious Grid applications, an infected or attacked Site's infrastructure and WNs or a VO's Central Grid Services, the presented mechanisms will ensure that no counterfeit evidence for a Grid job submitter's responsibility can be created. Generally, the utilization of File and Job Certificates will allow to limit a Site's and WN's permissions to the necessary minimum. Hence, File and Job Certificates can protect sincere users from false accusation and, if necessary, facilitate users to defend their reputation.

System integrity and confidentiality Due to the enforced authorization, limited permissions of VO- and Site-based services and verifiable interactions and processing, the presented mechanisms afford escalated protection against malicious behavior and both internal and external attackers, in particular in combination with the envisioned use of an isolation mechanism for Grid jobs during their execution. Together with mutually authenticated and encrypted connections, a Grid layer's system integrity can be protected with respect to illegitimate access (see Objective 4.2 and Objective 4.3) and interception and illicit information retrieval (see Objective 4.5).

Availability The presented mechanisms of Grid file system consistency, certified Grid files and on-demand storage discovery facilitate an assured and transparent handling of replicated data as well as of potential errors and failover. Together with the mechanism of certified Grid jobs, transparent processing and executions of Grid jobs are further achieved due to the the feasibility of verifiable rollbacks. In the event of any errors or outages, a failover of the Grid job execution as simply a anew submission of concerned Grid jobs becomes possible (see Objective 4.3).

8.5 Summary

This chapter presented a new security architecture for e-Science Grids established on the framework of *mediated definite delegation* (see Chapter 7), with user-signed *File Certificates* for Grid files and user- and subsequently service-signed *Job Certificates* for Grid jobs as conversions of the framework's authorship declaration and mediated delegation.

Within the security architecture, all communication via untrusted networks is required to be mutually authenticated and encrypted.

Preserving the e-Science Grid architecture's separation of logical Grid File Catalogs and distributed Storage Elements (SEs) for physical storage (see Chapter 7), the security architecture involves a Grid file access mechanism based on service-signed tickets in order to grant Grid users physical data access on SEs. In case of write operations, SE-signed tickets, replied by SEs to the Grid File Catalog's services via Grid users, are utilized in order to confirm authenticity of Grid file data during write operations together with user-signed File Certificates as declarations of authorship. Further on, an introduced booking table within the Grid file access mechanism assures consistency of transactions and enforceable authorization as well as transparent cancellation and rollback of Grid file operations. A dynamic discovery of SEs allows for the transparent utilization of data replication and failover as well as for load-balancing and automated data distribution.

In matters of Grid job submission, user- and service-signed Job Certificates as mediated delegations enable a transparent cancellation and rollback of Grid job processing and execution whenever necessary. Moreover, the security architecture and its Job Certificates prepare for the utilization of isolation mechanisms for Grid job execution.

The security architecture's File and Job Certificates afford integrity and authenticity of the respective entities and non-repudiation of their authorship as well as verifiable Grid job submissions and processing. Hence, meaningful decision-making with respect to the traceability of Grid files and jobs can be achieved.

As such and beyond the underlying framework of mediated definite delegation, the presented e-Science Grid security architecture facilitates the fulfillment of the security objectives defined in Chapter 4.5.

"All I really had was a suitcase and my drums. So I took them up to Seattle and hoped it would work."

Dave Grohl

9 A Grid Middleware Prototype

The presented security architecture (see Chapter 8) has been implemented to large extents as a prototype within the framework of a central Grid middleware project, called *jAliEn* [22]. This chapter will present and specify the prototype's implementation.

After a presentation of the secured communication implemented in the jAliEn Grid middleware prototype, its central communication protocol based on the exchange of serialized Java objects will be specified, which represents a core feature also with respect to security concerns. Further on, the prototype's design and arrangement of central and Site-based Grid services and the Grid client interfaces will be presented. Subsequently, the implementation of certified Grid Files and Jobs as a proof of concept will be specified, and finally, the capacity of the prototype will be demonstrated in a comprehensive performance evaluation.

Earlier versions of parts of this chapter have been presented in [PUB9, PUB12].

9.1 Secured Communication

All network-based communication of jAliEn Grid services is established via mutually authenticated and encrypted connections, utilizing the Transport Layer Security (TLS) [70] protocol with X.509 certificate-based authentication based on the Bouncy Castle Crypto API [130]. This includes a client-based service that establishes the connection to the Grid layer for every client system. This client-based service functions as a local service to actual client interfaces, while using a different local connection schema (see Section 9.4).

The mutual authentication of jAliEn Grid services is based on Grid user's X.509 certificates on part of the client-based service and X.509 host certificates for any other Grid services. Upon startup of a jAliEn service, an X.509 certificate and the corresponding key is loaded from a password-protected file container into a *Java Keystore* [106]. Accordingly an unprotected storage of loaded credentials, e.g. on a file system or a shell environment, can be avoided.

9.2 Serialized Java Object Streams

Within the mutually authenticated and encrypted connections, the communication of jAliEn Grid services is established exclusively by the exchange of serialized Java objects, analogous to Java Remote Method Invocation [107]. Further, the network-based communication of any pair of jAliEn Grid services is arranged via a single persistent connection, which is reestablished automatically whenever interrupted. Hence, all Grid services communicate via secured and persistent connections via the exchange of serialized Java objects. Thereby the exchange of serialized Java objects is restricted to objects of classes known to both communicating peers beforehand and the exchange of only object values, viz. no class definitions are exchanged.

API On top of this serialized object exchange, an application programming interface (API) within the Java program code handles code-internal requests for objects. All components operating on this API become completely independent of their environment, viz. the API is equally utilized within the different services of the Grid middleware including client- and Site-based services.

Dispatcher Below this API, a request dispatcher entity handles incoming requests based on the capacity of the respective process: in case the dispatcher entity is executed within a central Grid service it will directly process the request, while in the opposite case the request is sent to a central Grid service as a serialized object for remote processing. In both cases the dispatcher entity will return the respective response object of a request in identical manner, regardless if it was locally processed or received as a serialized object after remote processing. The dispatcher entity's behavior is triggered only within runtime of the program code via configuration parameters. In case it encounters the configuration of a non-central Grid service it automatically initiates the establishment of a connection to a given central Grid service, which potentially is itself a non-central Grid service. The latter is facilitated due to the universality of the API and the dispatcher entity, which allows a non-central Grid service to externally operate alike a central Grid service, and accept and handle requests from other Grid services, which are then sent for remote processing transparently. Presuming according secured connections, this design allows to concatenate several non-central Grid services, with only the 'last' one in the assemblage requiring a connection to a central Grid service in order to provide remote processing of requests for all non-central Grid services in the chain.

Request The central entity of the API is a *Request* class, which is inherited by all possible requests. The class handles generally relevant request attributes, for instance the username in favor of which the request is created as well as potential Java exceptions that occurred during the processing of a requests. In case a request is sent via object serialization to a remote service for processing it will further contain information of the requester's X.509 certificate, in order to facilitate e.g. verifications of identities or digital signatures. This information is provided by the dispatcher entity based on the established TLS connection via which the request is exchanged. Hence, remote requests can be implicitly authenticated based on the authentication of the respective connection and the therefore utilized X.509 certificates. Examples of requests utilized via the API as descendants of the Request class are the retrieval of a Logical File Name (LFN) object by an LFN string, the retrieval of an Access Ticket for an SE and the submission of a Grid job by passing on a signed JDL. Consequently, all conceptual Grid entities handled within the middleware, such as LFNs, Physical File Names or Grid Job JDLs, are implemented as Java classes and strict enforcement of authorizations and verifications can be easily achieved, using fine-grained controls on object attributes.

9.3 Central and Site-based Grid Services

The persistency back-end of jAliEn, with a central Grid File Catalog and Grid Task Queue, is an extension of the respective implementation within the AliEn middleware project (see Section 3.1), utilizing SQL databases. Further, the utilization of a central LDAP directory service and the according LDAP schema were adopted from AliEn.

Persistency The databases-established persistency back-end is fully encapsulated via another API within jAliEn. This low-level API facilitates the mapping of Grid entity objects within the middleware, such as objects of LFNs, Physical File Names or Grid Job JDLs, to database queries within the Grid File Catalog and Grid Task Queue. Hence, the jAliEn Grid middleware is apart from this API's components completely independent of its persistency back-end, which also represents an implicit protection against SQL or alike code injections by clients.

Services Within jAliEn, one central Grid service, named *jCentral*, provides central Grid layer access to client and Site services. The main Site service and Grid Site Agent, named *jSite*, is assumed to operate on a VO-Box and acts as a communication router for the jAliEn Grid Job Agent, called *jAgent*.

Service aggregation As the connection schema of all jAliEn Grid services is identical, jSite services can be optionally cascaded, allowing various levels of aggregation and routing (see Section 9.2). In particular both Site and client services (specified hereafter) can be equally aggregated via such service instances as connection routers, which represents a novelty and extension with respect to the service layout of the e-Science Grid architecture (see Section 3.3). Based on this aggregation and routing functionality it would be possible to establish federated structures or interconnections of Virtual Organizations (VOs). This has so far only been supported in the perspective of the connection and service schema, and has not further been implemented with respect to a view and domain separation of multiple VOs within the Grid data and job layer of an e-Science Grid. Moreover, the handling of requests in plaintext within the API and dispatcher entity must be considered, which as such allows no confidential information exchange via the routing Grid service.

9.4 Grid Client Interfaces

In order to provide for single sign-on access via Grid Client Interfaces, a client service entity named *jBox* is introduced. Upon startup on a Grid user's operating system, the user's X.509 certificate and the corresponding key are loaded and the jBox instance is dispatched as a process daemon.

jBox Once initialized, the jBox service immediately establishes a persistent connection to the jCentral service and sets up itself as a local service, listening on the operating system's loopback network interface and providing connectivity for Grid Client Interfaces via a line-based protocol. The service's port address and a generated authentication token are placed in a specified temporary file, protected from other operating system user's access via the file system's permissions.

jSh A Grid Client Interface, named *jSh*, provides a Grid user with command-line-based access to the Grid layer, while its functionality and command set is analogous to the AliEn Shell (see Section 3.1.2). The jSh interface can be started and stopped by a user independently of the local jBox service executed in background as a process daemon. Upon startup, jSh retrieves the jBox instance's port address and the authentication token from the temporary file and subsequently connects to the jBox service. The local connection between the jBox instance and the jSh client is unencrypted, and is consequently only protected by the resilience of the operating system's loopback network interface against potential listening attacks.

jRoot With respect to a potential adoption of the jAliEn Grid middleware within the ALICE experiment, an adapted version of the ROOT AliEn Interface (see Section 3.1.2), named *jRoot*, was developed as an interface library for the experiment's physics analysis framework AliRoot [6]. Analogous to the jSh interface, jRoot connects directly to the jBox instance and provides Grid layer access, e.g. to the Grid file system, for an analysis framework instance on a Grid user's system, or to a framework instance started within a Grid job (specified hereafter).

The jAliEn connection schema of all client-side entities and their connections is illustrated in Figure 9.1.

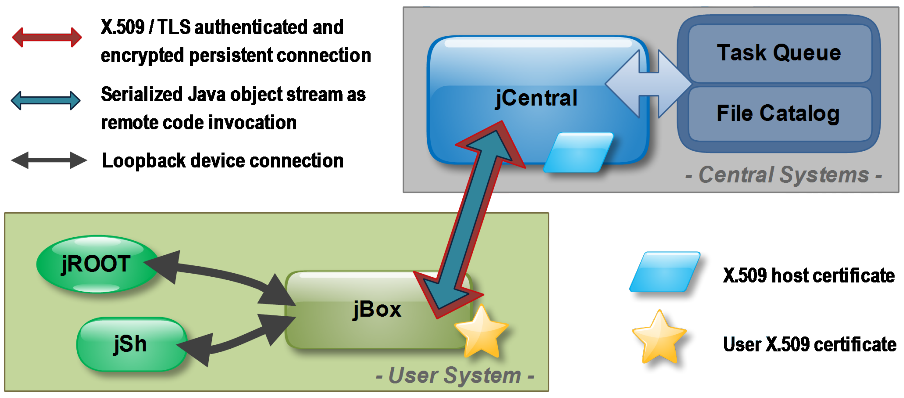

X.509 / TLS authenticated and encrypted persistent connection
Serialized Java object stream as remote code invocation
Loopback device connection
jCentral
Task Queue
File Catalog
- Central Systems -
jROOT
jSh
jBox
- User System -
X.509 host certificate
User X.509 certificate

Figure 9.1: jAliEn, connection of Grid Client Interfaces.

Interface independence The line-based protocol utilized by the jSh and jRoot interfaces for communication with the jBox service exchanges textual information only using a simple request and response scheme. The received textual requests are translated within the jBox service into object requests to the central API, with the consequently received response objects being again translated into textual replies to the interfaces. Hence, this protocol as well as the local interfaces are independent of the Java object exchange, and consequently of the Java programming language, and can be implemented independently. While the jSh interface is implemented in Java nevertheless, the jRoot interface is implemented as a C++ library in order to allow its utilization within the AliRoot framework.

9.5 Grid Data and Job Layer Security

The defined Grid security architecture (see Chapter 8) was implemented within the jAliEn Grid middleware prototype. With respect to architecture's mechanisms of Grid data layer security, the Booking Table, the SE access via Access and Status Tickets and the on-demand SE discovery are fully implemented. Consequently, the middleware provides full access to a Grid File Catalog with the physical data storage being established on provided SEs, which are accessible via the XRootD protocol. The implementation of Access and Status Tickets utilizes the *SHA384withRSA* scheme for digital signatures, provided by the Bouncy Castle Crypto API. The prototype implementation, however, does not include generation and usage of the defined File Certificates.

In matters of the Grid security architecture's mechanisms of Grid job layer security, the prototype implementation represents a functional proof of concept of certified Grid jobs. This implementation as well as envisioned functionality will be specified within the following paragraphs and is further illustrated in Figure 9.2.

Grid job submission Upon Grid job submission, the jBox service digitally signs the Grid job request using the Grid user's private key, corresponding to its X.509 certificate, and submits it to the jCentral service. There, the Grid user's signature and validity are verified before the job is authorized and placed into the Grid Task Queue. If not available already, the Grid user's certificate is stored in a central certificate database.

jAgent The jAgent service is developed as a Grid Job Agent within a pilot job model [42, 58], though there is so far no support for a submission of jAgent requests to a resource management system, and jAgent instances are required to be started manually. In matters of authentication and identification of a jAgent service instance, a random numerical identifier is utilized within the prototype implementation.

Provided with an X.509 certificate and a corresponding private key, a jAgent instance creates its numerical identifier upon startup and establishes a persistent connection to its dedicated jSite service. Subsequently, it requests Grid jobs via its jSite service connection from the jCentral service, while specifying the numerical identifier.

With respect to a future pilot job model, the creation of an identifier would be required to be further adapted to the submission of jAgent requests by the jSite service, as well as the supply of the X.509 certificate and a corresponding private key.

Grid job processing If a waiting job in the Grid Task Queue exists, the job's user-signed Grid job request is processed and digitally signed by the jCentral service, using the private key of the service's X.509 host certificate, resulting in a valid Job Certificate. Subsequently, the job is marked in the Grid Task Queue to be taken by the jAgent instance, referring to the provided numerical identifier, and the Job Certificate is sent together with the submitter's X.509 certificate to the jAgent service.

Grid job execution Once receiving the Job Certificate the jAgent verifies both digital signatures, while the X.509 host certificate used by the jCentral service is statically provided. In case of a successful verification, necessary Grid files are downloaded to the WN from an SE, the Grid job is executed by the jAgent and its output files are uploaded to an SE and registered in the Grid File Catalog. The read and write access within a jAgent instance utilizes Access and Status Tickets (see Section 8.1), the requests are though so far only authenticated and authorized based on the jAgent's identifier, viz. the Job Certificate is not used as a credential.

Job Certificate A simplified example of a Job Certificate with two applied signatures as utilized within jAliEn is given in Listing 9.1. The Job Certificate's digital signatures are based on the Bouncy Castle Crypto API, using the *SHA384withRSA* signature scheme.

For each digital signature, the job signature mechanism appends five signature tags as key-value-pairs to a JDL, or an already signed JDL. The key-value pairs *Signature_Issued* and *Signature_Expires* represent the timestamps for validity of the respective delegation. The *Signature_HashOrd* specifies the order the JDL key-value pairs appearance in the character string that is digitally signed. Hence, a JDL as well as a signed JDL or Job Certificate can be processed as a data object containing an unsorted set of key-value pairs without respecting their order. Further on, the *Signature_CertSerial* defines the serial number of the X.509 certificate utilized for a digital signature. Before its digital signature, the four key-value pairs of Signature_Issued, Signature_Expires, Signature_HashOrd and Signature_CertSerial are appended to a JDL's character string, as specified by Signature_HashOrd. Finally, the *Signature_SHA384withRSA* key-value pair with the digital signature of the signed JDL is appended.

```
OutputFile = "stdout,stderr,resource";
JDLPath = "/jalien/user/j/jay/testjob";
Executable = "/jalien/user/j/jay/bin/date";
Signature_Issued* = 1378527416;
Signature_Expires* = 1379737016;
Signature_HashOrd* = "Executable-[...]-Signature_CertSerial";
Signature_CertSerial* = "869671";
Signature_SHA384withRSA* = "mfgiWjT [...] FuJA8juK1Kzw==";
User = "jay";
Signature_Issued = 1378527420;
Signature_Expired = 1379737020;
Signature_HashOrd = "Executable-[...]-Signature_CertSerial";
Signature_CertSerial = "321957";
Signature_SHA384withRSA = "lsKwOrb [...] ntpSlBE0vAAQ==";
```

Listing 9.1: A sample Job Certificate (hashing orders and signatures truncated).

In case an initially signed Grid job request is to be signed a second time due to the creation of a File Certificate, the mechanism appends an asterisk '*' as a marker to the respective tags of the previous signature. The marked tags become part of the signed character string, thereby preventing a mix-up of tags of the two signatures. While new key-value pairs can be added to a JDL during its processing, key-value pairs existing in initial JDLs can also be modified (transformed) by the central Grid service as a broker. In this case, the initial key-value pairs are also marked by an asterisk, and new key-value pairs are appended additionally. Hence, it is possible to reproduce the initial form of a JDL or Grid job request and verify both the transformations applied and the corresponding first signature.

In contrast to the framework of mediated definite delegation (see Section 7.3), the implementations of the initially signed Grid job request and the File Certificate do so far not contain any identity of the broker or agent the respective request is assigned to. This feature can though be directly implemented based on the distinguished names (DNs) of the respective X.509 certificates, utilized by the jCentral and jAgent services. The DNs must then be appended as an according key-value pair to the Grid job requests before their signature.

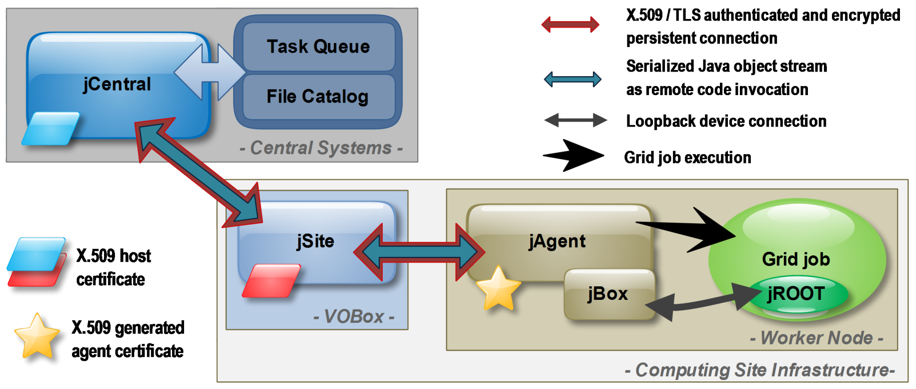

jCentral
Task Queue
File Catalog
- Central Systems -
X.509 / TLS authenticated and encrypted persistent connection
Serialized Java object stream as remote code invocation
Loopback device connection
Grid job execution
jSite
- VOBox -
jAgent
jBox
Grid job
jROOT
- Worker Node -
- Computing Site Infrastructure-
X.509 host certificate
X.509 generated agent certificate

Figure 9.2: jAliEn Site services: jSite, jAgent and jBox.

Grid job data access and Grid job isolation By an according extension of the prototype implementation, the jBox instance jRoot interface can further enable access to the Grid layer during a Grid job's execution. Therefore a jAgent service would be required to start a jBox instance before a Grid job is executed. Using the jRoot interface, an analysis framework instance within a running Grid job could then be provided with Grid layer access via the local jBox service, analogous to the scenario on a Grid user's system. In order to transparently separate accesses of different concurrently executed Grid jobs, a respective authentication token would be required for every Grid job executed, in order to allow the establishment of a connection of the jRoot interface and the jBox instance. Equal to the Grid file accesses by a jAgent, Grid file system accesses requested via the jRoot interface could then be authorized based on a legitimation within the respective Job Certificate.

In matters of Grid job Execution Protection (see Section 8.2.3), the WN service layout of jAliEn, with its jAgent and jBox services, and the functionality of the jRoot interface was designed in order to directly support scenarios involving mechanisms for Grid job isolation, such as gLExec [86] or virtualization: the Grid job execution is fully independent of the jAgent service and the connection of the jBox service and the jRoot interface is only based on the loopback network interface. In a scenario involving virtualization, this connection could be utilized via a virtual network interface within a respective virtual machine that is connected to the loopback interface of the physical host.

9.6 Performance Evaluation

In the following section a performance evaluation of the jAliEn Grid middleware prototype will be presented. In three different tests, the middleware's performance was evaluated, while two tests involved a comparison with the AliEn middleware.

The first performance test is a comparison of jAliEn and AliEn Grid Client Interfaces in terms of scalability based on concurrent Grid job submissions in an isolated test environment. Within the second test, the scalability of the jAliEn middleware is further analyzed in the same setup concerning concurrent Grid File Catalog directory listings as relatively simple, small and fast requests. The final third test is a study of the jAliEn middleware and the AliEn Grid Client Interfaces in terms of latency via an Internet-based connection and measures the response time of consecutive Grid File Catalog directory listing requests

Architectural comparison of AliEn and jAliEn The AliEn Perl Shell as well as the AliEn Shell (see Sections 3.1.2) utilize a string-based transmission of printout values. Hence, a client-side command, such as a directory listing within the Grid file system, is sent as a single request to AliEn central services. There the respective command's result is fully assembled, and, once sent to the client, directly printed out on the interface's screen. The connection is thereby established for each single request.

In contrast, service communication in jAliEn is based on the serialized Java object exchange protocol within a persistent connection, and requests are executed object by object with the screen printout being assembled by the client-side jBox service. Finally, the connections of AliEn Perl Shell, AliEn Shell and jBox with their respective Central Grid Services are based on different authentication and encryption mechanisms (compare Sections 5.1 and 9.1), while most importantly the AliEn Shell interface uses a proprietary authentication token and only client-to-service-side encryption. These architectural differences will be further considered within the results of the following tests.

9.6.1 Stress Test of Grid Job Submission

Concerning the potential performance impact of the Grid job signature mechanism of the jAliEn prototype implementation, and its scalability with respect to concurrent Grid job submission requests to its jCentral service, an according stress test was performed. In order to further allow for comparison, the stress test was equally applied to the AliEn middleware and its two Grid Client Interfaces, AliEn Perl Shell and AliEn Shell.

The tests were established in laboratory conditions, using three test machines: one machine (Intel Core 2 Duo CPU 2.80GHz, 8GB RAM, solid-state drive) was providing all necessary Central Grid Services, including required databases and a directory service, with both jAliEn and AliEn services deployed to operate on the same virtual Grid configuration, using the same Grid File Catalog and Grid Task Queue. Two other machines were utilized as clients, hereafter referred to as 'Client A' (Intel i5 CPU 2.50GHz, 8GB RAM, solid-state drive) and 'Client B' (Intel Core 2 Duo CPU 1.86GHz, 4GB RAM, solid-state drive). For all tests, an X.509 CA certificate with a 4096bit RSA public key and X.509 certificates with 2048bit RSA public keys were utilized for mutual authentication as well as for the signature of Grid job submissions in the jAliEn middleware. The signature in the jAliEn Grid job submission was based on the *SHA384withRSA* signature protocol via the *BouncyCastle* [130] provider.

For each of the Grid Client interfaces, different numbers of concurrent job submissions were independently tested as different test settings. In each setting the submissions were executed on both client machines side by side and the results were taken for exactly 300 seconds. For all three interfaces in all tested settings, the evaluation statistics were composed as the average, minimum and maximum response time and the count of successful job submissions, and the according average frequency rate of requests that was reached by the two clients.

In all tests the same minimal Grid job JDL was submitted, with a size of 79 bytes, specifying only an executable and output files, while the JDL content was directly provided as an input parameter to the command. The measurement referred to the request-to-response wall time of a job submission, viz. the time of submission till the service states its successful placement in the Grid Task Queue by replying an identifier of the queued job. The time measurements were taken in all three implementations internally via adaptions in the respective program code, while in JAliEn the measurements were taken directly within the jBox service. Hence, the line-based local connection to a jSh instance over the loopback network device was not included.

	Average:	**Min.:**	**Max.:**	**Count:**	**Freq.:**
20 submission threads per client (2p10t)					
Client A	22ms	13ms	284ms	28068	94Hz
Client B	28ms	20ms	467ms	21489	72Hz
20 submission threads per client (10p2t)					
Client A	71ms	15ms	742ms	42172	141Hz
Client B	125ms	28ms	849ms	23629	79Hz
1000 submission threads per client (20p50t)					
Client A	147ms	16ms	1s	42106	140Hz
Client B	282ms	43ms	1.5s	20159	67Hz
1000 submission threads only on Client A (20p50t)					
Client A	146ms	21ms	879ms	41136	137Hz
Client B	-	-	-	0	0.00Hz
2000 submission threads per client (20p100t)					
Client A	143ms	20ms	845ms	41162	137Hz
Client B	282ms	64ms	1.3s	20223	67Hz

Table 9.1: jAliEn, concurrent submissions of signed Grid jobs in loop per client.

jAliEn The jAliEn Grid Client Interface was tested in different settings of 20, 100 and 2000 concurrently processing loops of submission requests on the two independent clients. The looped concurrent submissions were established by a number of jBox service instances, with inside each instance a number of internal Java-based threads continuously issued job submissions. Due to the architecture of jBox, the internal threads of a jBox instance (hereinafter referred to as a process) shared one persistent connection to the jCentral service within each setting. The different test settings are therefore additionally marked with an according number of jBox processes and internal Java-based threads. Table 9.1 presents the statistics of the different settings of the jAliEn middleware.

For 20 concurrent Grid job submissions in loop per client, the results show a dependency with respect to the setup of processes and threads: in the first setting, each client executed 2 processes and 10 threads (2p10t) and had consequently two connections to the jCentral service. In the second setting, each client executed 10 processes and 2 threads (10p2t) and had consequently ten connections to the jCentral service. The response times in the first setting are considerably lower, which could be explained by a pooling of threads with respect to the number of connections. At the same time the total number and frequency of submissions is though significantly lower, which cannot be explained and is simply taken as an identified dependency.

Figure 9.3 and Figure 9.4 show the according histograms for these settings, and Figure 9.5 shows a histogram of 1000 concurrent Grid job submissions in loop.

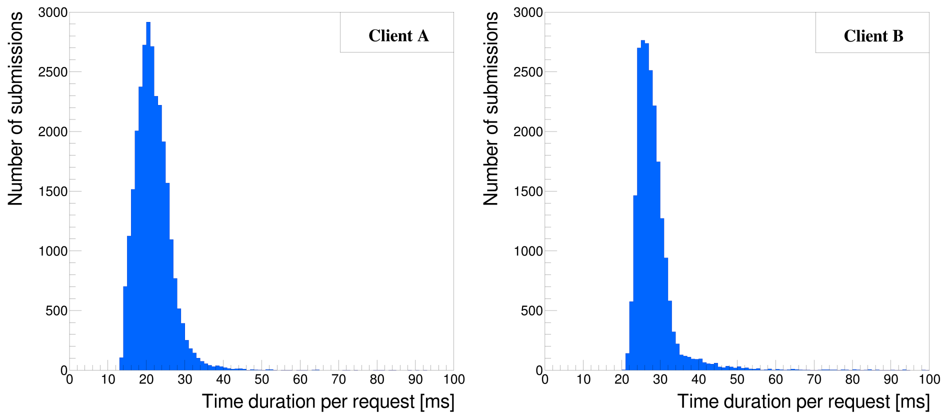


Figure 9.3: jAliEn jSh: Two clients with each 20 concurrent Grid job submissions in loop (2p10t).

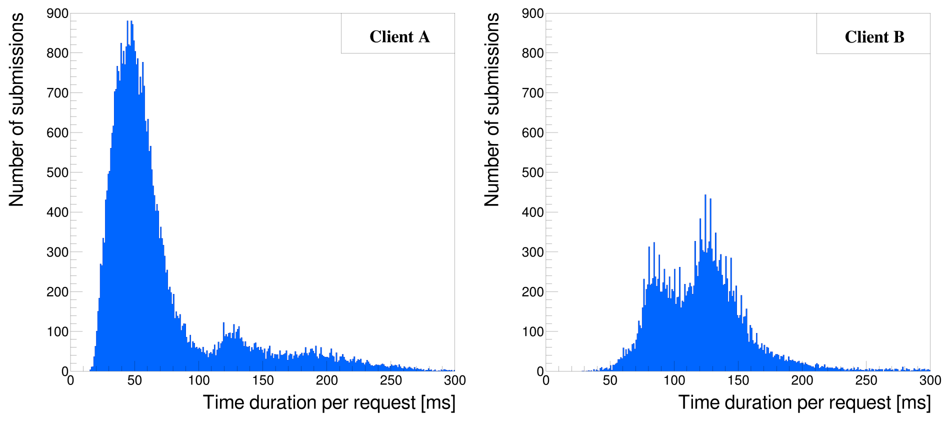


Figure 9.4: jAliEn jSh: Two clients with each 20 concurrent Grid job submissions in loop (10p2t).

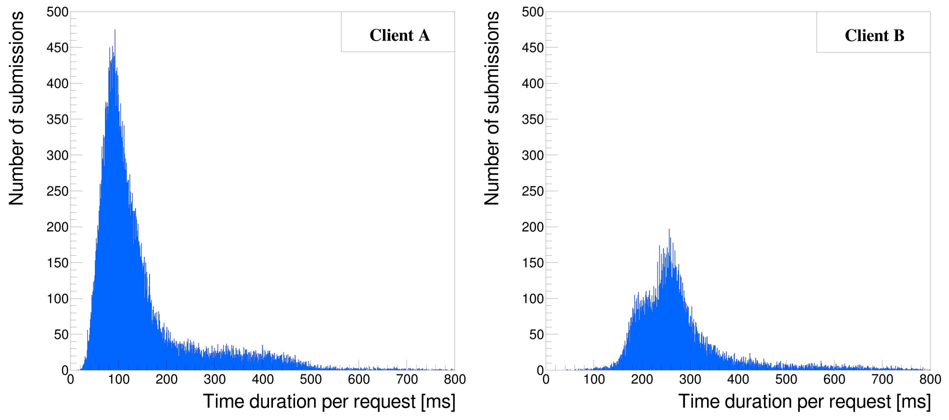


Figure 9.5: jAliEn jSh: Two clients with each 1000 concurrent Grid job submissions in loop (20p50t).

The settings of 20, 1000 and 2000 concurrent Grid job submissions in loop per client show a correlation in total number and frequency of submissions. This correlation was further confirmed by a setting involving only 1000 concurrent Grid job submissions by client A, while client B was excluded in this setting: the highest frequency measured with client A is approximately at 140 submission requests per second, even if the jCentral service was able to process more than 200 submission requests per second in settings involving both clients. Hence, it was obviously not possible to achieve an overloading of the jCentral service by the presented settings, and the clients reached a certain limitation. This limitation is assumed to be related to the computationally more expensive digital signature of the job request on client-side, in comparison to the signature's service-side verification. The client limitation as such was further confirmed by a corresponding performance test of Grid File Catalog access, presented hereafter in Section 9.6.2.

In summary, the client-side limitation allowed for a frequency up to more than 200 verified and accepted Grid jobs per second by the jCentral service (combined Client A and B). Thereby the average response time remained below 300 milliseconds in all test settings, while the worst maximum response time measured was at 1.5 seconds. In order to identify a potential influence of the digital signature of a job request within a submission, two test settings involved the disabling of the job signature, viz. the job requests were submitted only as plain and unsigned JDL listings. The statistics of these two settings are presented in Table 9.2, in which the frequency of submissions per second increased with a factor of roughly two to three in comparison to the test settings of signed submissions. The maximum response times were relatively higher, which is assumed to be a result of a relatively increased load on the jCentral service indicated by the higher frequency. Finally, the average response time of unsigned submissions were lower than the respective signed submissions again with a factor of approximately two to three.

	Average:	**Min.:**	**Max.:**	**Count:**	**Freq.:**
20 unsigned submission threads per client (2p10t)					
Client A	11ms	4ms	659ms	53784	179Hz
Client B	12ms	5ms	663ms	51357	171Hz
1000 unsigned submission threads per client (20p50t)					
Client A	71ms	5ms	1.4s	86198	287Hz
Client B	94ms	6ms	6.6s	59553	199Hz

Table 9.2: jAliEn, concurrent unsigned Grid job submissions in loop per client.

AliEn The two AliEn Grid Client Interfaces *AliEn Perl Shell* and *AliEn Shell* (see Section 3.1.2) were tested in three independent settings of 1, 10 and 20 concurrent Grid job submissions in loop on each client.

Table 9.3 presents the statistics of the three tested settings of the AliEn Perl Shell.

	Average:	Min.:	Max.:	Count:	Freq.:
1 submission thread per client					
Client A	365ms	158ms	16.3s	791	2.64Hz
Client B	379ms	158ms	16.3s	771	2.57Hz
10 submission threads per client					
Client A	3.0s	233ms	111.4s	751	2.50Hz
Client B	3.0s	408ms	84.1s	636	2.12Hz
20 submission threads per client					
Client A	8.4s	370ms	249.5s	294	0.98Hz
Client B	28.2s	1.1s	248.9s	76	0.25Hz

Table 9.3: AliEn Perl Shell, concurrent Grid job submissions in loop per client.

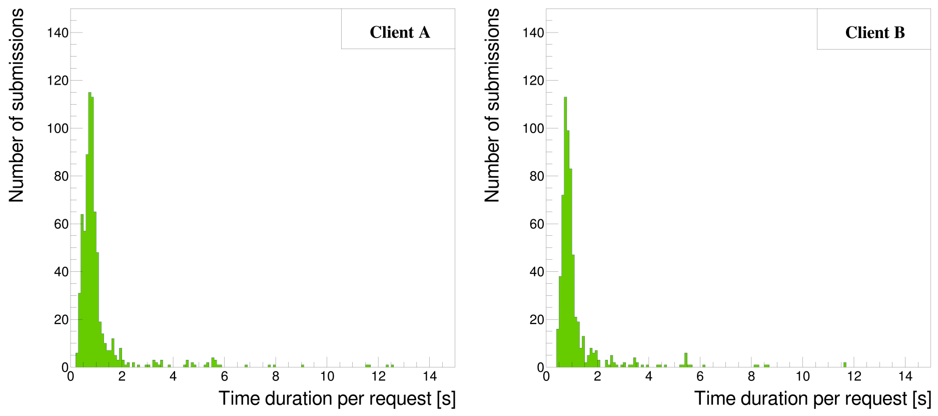


Figure 9.6: AliEn Perl Shell: Two clients with each 10 concurrent Grid job submissions in loop.

At one concurrent Grid job submission in loop per client, the system was able to perform roughly two and a half submissions per second. While the average response time was below 400 milliseconds, the maximum response time was approximately 16 seconds. At 10 concurrent Grid job submissions in loop per client, the average response time escalated with the factor of 10 with respect to the previous setting and the maximum response time was beyond one minute. Figure 9.6 shows the results of this setting.

In the setting of 20 concurrent Grid job submissions in loop per client, the testing identified a strong overloading of the middleware and an according depletion in performance.

The statistics of the three test settings of the AliEn Shell are shown in Table 9.4. At one concurrent Grid job submission in loop per client,

	Average:	Min.:	Max.:	Count:	Freq.:
1 submission thread per client					
Client A	266ms	80ms	4.1s	967	3.22Hz
Client B	265ms	79ms	3.9s	956	3.19Hz
10 submission threads per client					
Client A	2.7s	134ms	39.3s	1107	3.69Hz
Client B	2.7s	222ms	32.5s	1084	3.61Hz
20submission threads per client					
Client A	4.8s	150ms	35.5s	1213	4.04Hz
Client B	4.7s	232ms	35.4s	1221	4.07Hz

Table 9.4: AliEn Shell, concurrent Grid job submissions in loop per client.

the system was able to perform roughly three submissions per second, with an average response time below 300 milliseconds and a maximum response time of approximately 4 seconds. The average response time scaled with the number of concurrent Grid job submissions in loop across the three settings, as did the maximum response time from 1 to 10 concurrent Grid job submissions. The frequency was increased to approximately four submissions per second in the setting of 20 concurrent Grid job submissions in loop per client. Figure 9.7 shows the results of this setting. Also for the case of the AliEn Shell, the test revealed a strong overloading of the middleware and an according depletion in performance.

Considering the upper frequency rates of signed job submission in jAliEn, the jCentral service was able to verify and accept more than 200

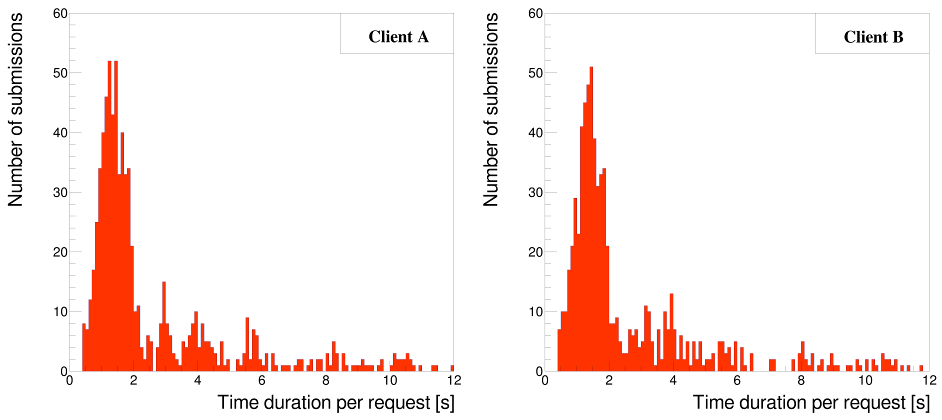


Figure 9.7: AliEn Shell: Two clients with each 20 concurrent Grid job submissions in loop.

submissions per second within these tests on standard hardware. Given the average response time of any measured signed job submission below 300 milliseconds and the mechanism's robustness in terms of scalability, the identified impact factor of two to three of the digital signature processing is assessed to represent no fundamental disadvantage in performance.

In case of the second signature applied to a Grid job request upon propagation to a Site as a File Certificate, the computationally more expensive digital signature would be applied by the jCentral service and the verification would be conducted by jSite services. Accordingly, the presented test results are assumed to apply reciprocally as a lower limit, as the test involved the concurrent generation of requests on client side, each requiring a digital signature.

In conclusion, considering the measured response times and frequencies and the identified behavior in terms of scalability of the AliEn Grid Client Interfaces, the jAliEn prototype revealed a considerable performance potential within this testing. In matters of a comparison with the AliEn middleware, the jAliEn middleware exceeded by large all reference values presented by the AliEn middleware implementations, despite its digitally signed submissions.

9.6.2 Stress Test of Grid File Directory Listing

Within a second scalability and stress test, the performance of Grid File Catalog accesses was evaluated based on directory listing requests.

The following results refer to the same testing setup as presented in the previous Section (see Section 9.6.1), while only involving the jAliEn middleware and replacing the Grid job submission by a directory listing request on a directory containing 20 file entries. Again, the requests where issued by the two clients with different numbers of independent request loops and the results were taken for 300 seconds.

The statistics of the different settings of 20, 100, 500 and 2000 concurrently processing loops of requests on each client are presented in Table 9.5. Figure

	Average:	Min.:	Max.:	Count:	Freq.:
20 ls request threads per client (2p10t)					
Client A	7ms	3ms	181ms	90458	302Hz
Client B	9ms	5ms	177ms	33740	113Hz
100 ls request threads per client (2p50t)					
Client A	9ms	4ms	860ms	66677	222Hz
Client B	10ms	5ms	865ms	60587	202Hz
100 ls request threads per client (2p50t) memory opt.					
Client A	9ms	4ms	622ms	67195	224Hz
Client B	10ms	5ms	622ms	60939	203Hz
500 ls request threads per client (10p50t)					
Client A	42ms	4ms	996ms	72301	241Hz
Client B	44ms	5ms	992ms	60763	203Hz
500 ls request threads per client (5p100t)					
Client A	21ms	4ms	1311ms	74691	249Hz
Client B	21ms	5ms	687ms	59211	197Hz
500 ls request threads only on Client A (5p100t)					
Client A	12ms	4ms	712ms	129358	431Hz
Client B	-	-	-	0	0Hz
2000 ls request threads per client (20p100t)					
ClientA	85ms	4ms	1234ms	73345	245Hz
ClientB	93ms	5ms	1347ms	59557	199Hz

Table 9.5: jAliEn, concurrent threads of looped requests for directory entries per client.

9.8 and Figure 9.9 show further the results of the test settings of 100 and 1000 concurrent request loops per client. In all test settings, the jCentral responded to directory requests (for both clients combined) at a rate above 400 replies per second. Furthermore, in all but the last setting (20p100t), the average response time was below 50 or even below 20 milliseconds, and for 2000 concurrent requests per client the average response time was below 100 milliseconds. Finally, the worst maximum response times measured were below 1.5 seconds, and considerably lower in majority of settings.

The test was able to confirm the client dependency of the previous test of concurrent job submissions by another setting involving only one client: within this setting, an expected overtake of the spare performance of the jCentral service with respect to only one client was identified, indicated by the respective total count and frequency of responded requests.

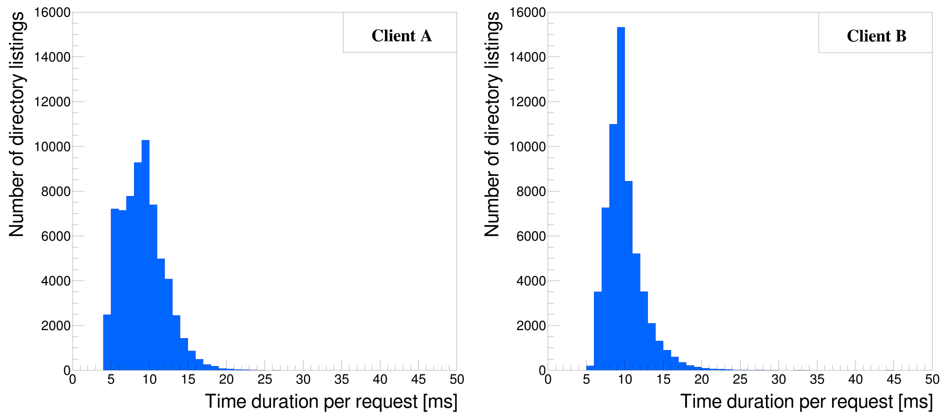


Figure 9.8: jAliEn jSh: Two clients with each 100 concurrent directory requests in loop (2p50t).

The yearly average rate of Grid file accesses (not directory listings) in the ALICE Grid Services, measured as authorization requests to central services, is approximately 230Hz [5], while corresponding daily average rates at approximately 400Hz were measured during execution periods of approximately 50k Grid jobs. Considering this as a reference point for a requirement of performance, the jAliEn prototype implementation was able

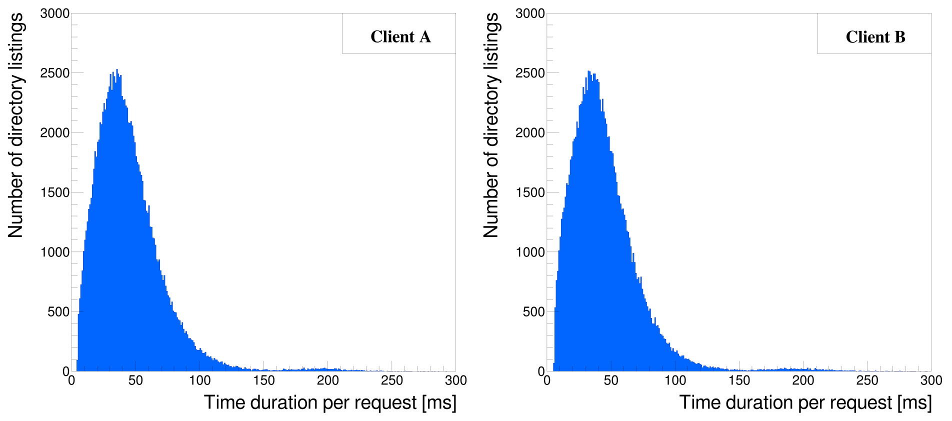


Figure 9.9: jAliEn jSh: Two clients with each 500 concurrent directory requests in loop (10p50t).

to provide such rates for simple requests while operating on common and non-specialized machines.

9.6.3 Latency Test of Grid File Directory Listing

The following test represents a comparison of the jAliEn prototype implementation and the AliEn middleware, arranged by an instantiation of both middlewares within the ALICE Grid Services in production. The test assessed the client-to-service response time for Grid File Catalog accesses in terms of network latency and Internet-based connections.

The setup involved the in-production instance of the ALICE Grid Services with its central services running at CERN (near Geneva) on part of the AliEn Grid middleware, and a jAliEn jCentral service deployed on a test machine at CERN, directly connected to the internal network of the ALICE Central Grid Services. The jCentral service was further provided with access to the in-production ALICE Grid File Catalog via a direct connection to the ALICE directory server and database system back-end.

The test involved two different client-side locations: in one case, the AliEn Perl Shell and AliEn Shell interfaces and the jBox service were executed on a test machine at CERN, directly connected to the internal network of the ALICE Central Grid Services, as a scenario with least-possible network latency and no external network interference. In the other case, the two interfaces and the jBox service were executed on a personal computer

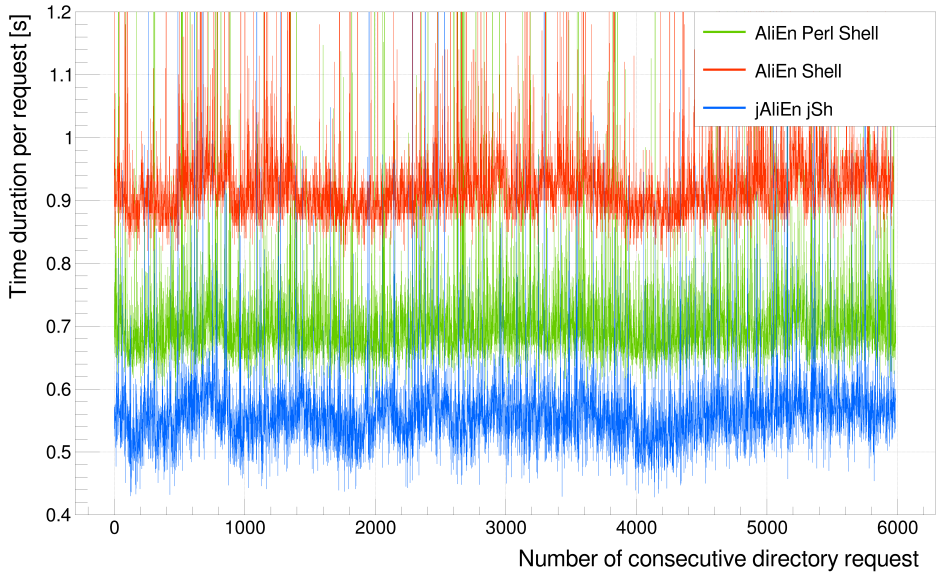


Figure 9.10: Berlin-to-CERN, detailed directory listing request [ls -l] response times.

located in Berlin, while using a non-permanent Internet connection. This second case was chosen in order to simulate a characteristic scenario of an external Grid user connecting via the Internet, while incorporating latency and Internet-traffic interference.

The test was further based on the two requests of a short and a detailed listing of the same Grid File Catalog directory. The listings were requested by executions of the commands [ls] and [ls -l] in each of the three implementations and had a resulting screen printout size of approximately 7k bytes for the short and approximately 40k bytes for the detailed directory listing. The measuring was based on the internal wall time, measured within the three implementations' program code, as the request-to-response time including the preparation of the respective screen output, viz. the second timestamp was taken before the output was sent to the screen for printout.

For both locations, the two requests were executed independently in bunches of 100 cycles, with one cycle being a consecutive timing of each of the three Grid Client Interfaces. This setting was repeated at various times across several days, in order to assure variations concerning the system load of the in-production ALICE Grid services and the Internet-traffic interference. The single bunches of the resulting four test cases were later concatenated and analyzed in total. The combined results of the Berlin-based detailed directory requests are shown in Figure 9.10, which are based on 60 concatenated bunches and consequently 6000 requests by each of the Grid Client Interfaces. The corresponding results of CERN-based requests are shown in Figure 9.11, which are based 120 concatenated bunches and consequently 12000 requests.

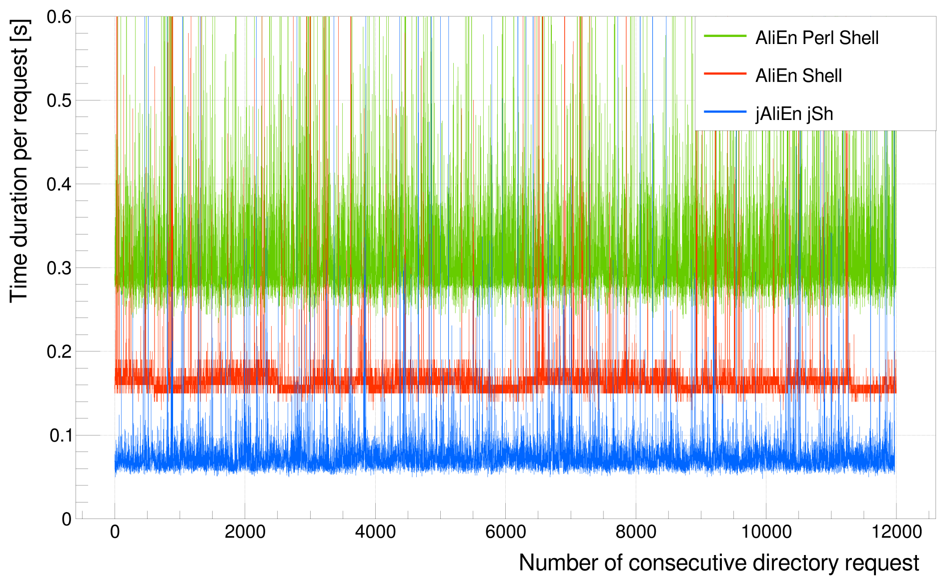


Figure 9.11: CERN-to-CERN, detailed directory listing request [ls -l] response times.

The statistics of all the four cases of short and detailed directory listings for CERN- and Berlin-based requests were calculated based on the combined bunches as the average, minimum and maximum response time and the number of request errors, with the errors not considered within the preceding calculations. Table 9.6 shows the statistics of the Berlin-based requests, and Table 9.7 the corresponding statistics of CERN-based requests.

Critical assessment The test compared two interfaces and respective services of a loaded in-production system with an interface connected to an

	Average:	Minimum:	Maximum:	Errors:
Short directory listing request [ls] statistics				
AliEn Perl Shell	638ms	522ms	5.7s	1
AliEn Shell	717ms	640ms	3.1s	7
jAliEn jSh	553ms	419ms	1.9s	3
Detailed directory listing request [ls -l] statistics				
AliEn Perl Shell	722ms	578ms	26.9s	11
AliEn Shell	931ms	810ms	5.9s	28
jAliEn jSh	567ms	428ms	1.7s	13

Table 9.6: Berlin-to-CERN, directory listing request statistics.

	Average:	Minimum:	Maximum:	Errors:
Short directory listing [ls]:				
AliEn Perl Shell	291ms	205ms	11,0s	0
AliEn Shell	144ms	110ms	2.0s	2
jAliEn jSh	77ms	46ms	1.3s	16
Detailed directory listing [ls -l]:				
AliEn Perl Shell	324ms	231ms	4.4s	0
AliEn Shell	169ms	130ms	1.8s	0
jAliEn jSh	78ms	48ms	1.2s	26

Table 9.7: CERN-to-CERN, directory listing request statistics.

otherwise idle service, which needs to be considered as a general limitation for its significance. Nevertheless, the interdependence and impact of the commonly used instance of the ALICE Grid File Catalog, and the respective database back-end, as well as any network-related conditions or influences accounted equally for all three cases. Moreover, the directory listing was chosen as a basic and relatively fast request, and the measurements were distributed over several days, in order to potentially capture time frames of different levels of load.

Evaluation and interpretation Concerning the results for the jAliEn middleware only, the average response time for Berlin-based requests was below 600 milliseconds for both request types, and the corresponding average response time for CERN-based requests was below 80 milliseconds. The maximum response time for Berlin-based requests was below 2 seconds and below 1.3 seconds for CERN-based requests. The consistency of the results of short and detailed requests was anticipated, as due to the serialized Java object exchange protocol the two requests account for the same Grid File Catalog objects to be transmitted to the client. Concerning the high error rates of the jAliEn interface, a synchronization bug of Java threads in the implementation of the jBox service was identified. The bug remained unfixed during the testing and is assumed to have caused at least a fraction of the measured errors.

An unexpected result of the tests are the average rates of the AliEn Shell interface for Berlin-based requests. Although the interface only uses one-side encryption and had, compared with the AliEn Perl Shell interface, relatively lower rates at CERN-based requests as expected, the rates were relatively higher for Berlin-based requests. Due to the identical measuring framework and the consistent observation of this result for both short and detailed directory listings, it is suspected to be caused by the AliEn Shell interface's protocol with respect to latency or its packets' Internet-based treatment, yet this finding was not further investigated.

The in-production state of the ALICE Grid Services is assumed to have only a small or potentially even negligible effect with respect to the one-time best measured response time of each of the two AliEn interfaces. Hence, these measured best response times can at least be considered as a general reference point with respect to the target-setting and performance requirements of the ALICE experiment, as they represent the performance of an actual system in place. Considering a comparison of the average response times of the jAliEn interface with the minimum response times of the two AliEn Grid Client Interfaces, the numbers are roughly equal for the case of Berlin-based requests and below in case of CERN-based requests.

In conclusion, the test showed no fundamental performance deficits of the jAliEn communication protocol with respect to latency, despite its exchange of high-level Java objects and an underlying secured connection. In matters of a comparison with the AliEn middleware, the jAliEn middleware was able to clearly exceed the reference values presented by the AliEn middleware implementations.

9.7 Summary

In this chapter, a prototype implementation of the security architecture for e-Science Grids (see Chapter 8) has been presented.

The prototype implementation within the jAliEn Grid middleware project affords secured communication based on mutually authenticated and encrypted connections utilizing X.509 certificates and the Transport Layer Security protocol. Its service communication protocol based on serialized Java objects facilitates the exchange of complex high-level objects and an according enforcement of access controls. A client-side service enables a single sign-on capability of respective client interfaces. Utilized on a WN this service functionality could further provide Grid layer access to Grid jobs during their execution. Thereby, additional mechanisms for Grid job isolation, such as gLExec [86] or virtualization, could be directly applied.

The prototype implementation represents a proof of concept of the previously presented e-Science Grid security architecture (see Chapter 8), most importantly by its implementation of the mechanism of certified Grid jobs. Its feasibility and capabilities were confirmed within the presented performance evaluation, revealing a notable performance potential with respect to service response times as well as to scalability and high loads.

"The only reason for time is so that everything doesn't happen at once."
Albert Einstein

10 Conclusion

E-Science Grids provide the researchers of multiple and possibly otherwise independent and globally distributed organizations with unified access to large-scale computing and storage services. It is part of their purpose that e-Science Grids allow collaborating researchers as users to introduce their own data and program code in the course of their work. Beyond, via submission of Grid jobs any program code can be requested to be executed as detached computational operation within the distributed computing infrastructure. The delegation of privileges in the course of Grid jobs submissions in combination with the users' allowance to introduce and utilize a priori untrusted program code and data is though a widely identified security challenge [78, 61, 121]. In response to this challenge, a new framework for delegation and an according Grid security architecture have been presented in this thesis.

On the basis of a study and comparison of the ALICE Grid Services as a globally distributed e-Science Grid and other e-Science infrastructures and approaches, a generic e-Science scenario has been defined. Further on, the architecture of the in-use ALICE Grid services has been analyzed and compared with architectures of other existing e-Science Grids. Based on this comparison, a generic e-Science Grid architecture has been defined.

In reference to the defined e-Science scenario and Grid architecture, the involved players and their relationships, the security characteristics and potential attack motives and targets have been specified, and according security objectives have been defined.

In matters of in-use e-Science Grids and corresponding security mechanisms, selected vulnerabilities of the ALICE Grid Services have been identified. Further on, the concept of delegation and its utilization in Grid computing as well as the deficiencies of unrestricted delegation based on X.509 proxy certificates have been analyzed and consequent vulnerabilities have been examined and specified. Finally, further alternative approaches for delegation and their capabilities were discussed.

Subsequently, a new delegation framework, named *mediated definite delegation*, facilitating transparent, dynamic and least-privileged delegation has been presented. The framework is based on textual statements for both task delegations and attestation of data authorship, which are required to be signed using public-key signature mechanisms. In case of task delegations, cascaded signatures are utilized within the different steps of submission and processing and propagation, while a task delegation can be processed and propagated via more than one broker to an agent for execution. The mechanisms afford non-repudiation of authorship of data as well as of task delegations and their processing and propagation. Moreover, task delegations as credentials afford protection against misuse of the delegating user's identity as well as against unnoticed alteration of the requested actions. The framework and its mechanisms integrate completely into the defined e-Science Grid architecture and require no remote callbacks or additional invocations.

Established on the delegation framework and including further mechanisms, a new security architecture for e-Science Grids has been presented. The security architecture introduces user-signed *File Certificates* for Grid files and user- and service-signed *Job Certificates* for Grid jobs as declarations of authorship and statements of mediated delegation. It provides enforced authorization for Grid layer access including storage-confirmed authenticity of Grid file data and arranges for transparent cancellation and rollback of Grid file and job operations. A dynamic discovery of physical storage devices allows for the transparent utilization of data replication and failover as well as for load-balancing and automated data distribution. File and Job Certificates afford integrity and authenticity of the respective entities and non-repudiation of their authorship as well as verifiable Grid job submissions and processing. Further on, the introduced Job Certificates prepare for the utilization of isolation mechanisms for Grid job execution. Beyond the underlying framework of mediated definite delegation, the presented security architecture thereby facilitates the fulfillment of the defined security objectives for e-Science Grids.

Finally, a prototype implementation of the new security architecture has been presented as a proof of concept. An evaluation of the implementation's performance has revealed notable capacity with respect to service response times as well as to scalability and high loads.

10.1 Future Work

The following paragraphs will outline potential future research related to the work and results presented in this thesis.

Standardization The presented security architecture for e-Science Grids and its mechanisms of File and Job Certificates rely on necessary definitions and specifications within an infrastructure or environment. Concerning the presented mechanism of Job Certificates, in particular the specification and fixation of utilized data formats as well as allowed processing and transformations have to be completed within the scope of an actual e-Science Grid and the respective Virtual Organization. By standardization of data formats and allowed transformations the mechanisms could be employed beyond organizational borders of one or a dedicated set of e-Science Grids. Hence, Job Certificates could be generally acknowledged as credentials for authentication and authorization within a distributed infrastructure, and thereby render the maintaining and, even more important, the control of agent instances, such as the Site-based Grid Job Agent, by brokers or VOs unnecessary.

Confidentiality of Grid files and jobs In both [112] as well as [45], Grid security architectures providing confidential propagation of Grid data and jobs based on public-key encryption are presented. In essence, a Grid job and necessary data are encrypted, using the public key of a service endpoint provider or Site, and are decrypted by the provider or Site before the job's execution, using its private key. Accordingly, a submitted Grid job as well as relevant data can be securely propagated from a user to a service endpoint provider via intermediate processors.

This mechanisms could be combined with the herein presented work, briefly outlined as follows: The mechanisms of both File and Job Certificates could be directly applied to encrypted data and Grid job specifications, while thereby equally providing authenticity, authorship and non-repudiation. Encrypted Grid job requests could be signed and submitted by users, while the Central Grid Services could process and propagate these job requests and 'blindly' apply their additional signatures. Hence, an endpoint provider would be able to verify the authenticity of the original submissions, the submitters' identity, as well as the authenticity of intermediate processing.

Virtualization and trusted computing In matters of Grid job execution protection, virtualization has been identified (see Section 8.2.3) as a strong mechanism for mutual isolation of Grid jobs as well as for the protection of a Grid Job Agent against access from executed Grid jobs.

By usage of trusted computing environments, such as proposed by the *Intel Trusted Execution Technology* [84], even stronger levels of isolation could be provided, while also lowering the necessary trust in a resource provider and integrity of execution environments [121]. Within such a scenario, mechanisms like the presented File and Job Certificates could equally apply, and afford verifiable authenticity and authorship of Grid files and job submission, while the according verifications could be provided within the trusted environment. Hence, all players, viz. users, brokers, and executing agents, could equally rely on the verification, while only depending on the trusted environment's operation and its manufacturer.

10.2 Cloud Computing

In matters of relevance of the herein presented work for the area of cloud computing, the Cloud service *Amazon Web Services (AWS)* [7] is an example for comprehensive customer liabilities, including theoretically severe consequences for users of the service.

An excerpt from the *Amazon AWS customer agreement* [10] is as follows: *"You will defend, indemnify, and hold harmless us, our affiliates [..] from and against any claims, damages, losses, liabilities, costs [..] arising out of or relating to any third party claim concerning: [..] your or any End Users' use of the Service Offerings [..] or violation of applicable law by you or any End User [..] Your Content or the combination of Your Content with other applications [..] a dispute between you and any End User. If we or our affiliates are obligated to respond to a third party subpoena or other compulsory legal order or process described above, you will also reimburse us for reasonable attorneys' fees, as well as our employees' and contractors' time and materials spent [..] at our then-current hourly rates."*

In a nutshell, this signifies a service provider deploying its services on the infrastructure of a cloud provider will be held liable for any consequences of the usage, including the usage of its own end users. Assuming a cloud-based e-Science infrastructure, this would imply a VO to be held liable for any consequences of the use of the cloud services.

Hence, the accurate identification of liability with respect to the usage of cloud services is a primal concern in case of security or legal incidents. Which includes the protection of sincere users from malicious misuse of their digital identity and consequent false accusation.

Mechanisms like a certification of the data imported and the tasks delegated by cloud users, analogous to the work presented within this thesis in the context of e-Science Grid infrastructures, are assumed to be able to thereby enhance, simplify and accelerate the identification of liabilities.

Bibliography

[1] ALICE - A Large Ion Collider Experiment. `http://aliceinfo.cern.ch/`.

[2] ALICE Collaboration. `http://aliceinfo.cern.ch/Collaboration/`.

[3] ALICE Grid Monitoring with MonALISA. `http://pcalimonitor.cern.ch/`.

[4] AliEn. `http://alien2.cern.ch/`.

[5] Alimonitor - Cache statistics. `http://alimonitor.cern.ch/display?page=machines/caches`.

[6] AliRoot Documentation - ALICE Offline Pages. `http://aliweb.cern.ch/Offline/AliRoot/Manual.html`.

[7] Amazon Web Services, Cloud Computing: Compute, Storage, Database. `http://aws.amazon.com/`.

[8] ATLAS Experiment. `http://atlas.ch/`.

[9] ATLAS Experiment Collaboration. `http://atlas.web.cern.ch/Atlas/Collaboration/`.

[10] AWS Customer Agreement. `http://aws.amazon.com/agreement/`.

[11] CBM - The Compressed Baryonic Matter experiment. `http://www.fair-center.eu/for-users/experiments/cbm.html`.

[12] CMS - Compact Muon Solenoid experiment at CERN's LHC. `http://cms.cern.ch/`.

[13] CMS - People Statistics. http://cms.web.cern.ch/content/cms-collaboration/.

[14] CMS - People Statistics. http://cms.web.cern.ch/content/people-statistics/.

[15] Condor High Throughput Computing - Classified Advertisements. http://research.cs.wisc.edu/condor/classad/.

[16] e-BioGrid. http://www.e-biogrid.nl.

[17] EGI - European Grid Infrastructure. http://www.egi.eu/.

[18] European Organization for Nuclear Research - CERN. http://www.cern.ch/.

[19] Extensible Markup Language (XML) 1.1 (Second Edition). http://www.w3.org/TR/xml11/.

[20] GridSite - Products - European Middleware Initiative. http://www.eu-emi.eu/products/-/asset_publisher/1gkD/content/gridsite-1/.

[21] International Grid Trust Federation - IGTF. http://www.igtf.net/.

[22] jAliEn. http://jalien.cern.ch/.

[23] LHCb - Large Hadron Collider beauty experiment. http://lhcb-public.web.cern.ch/lhcb-public/.

[24] mod_ssl: The Apache Interface to OpenSSL. http://www.modssl.org/.

[25] MonALISA - MONitoring Agents using a Large Integrated Services Architecture. http://monalisa.cern.ch/.

[26] MySQL. http://www.mysql.com/.

[27] National e-Science Centre definition of e-Science. http://www.nesc.ac.uk/nesc/define.html.

[28] Open Science Grid. https://www.opensciencegrid.org/.

[29] SURFsara. `http://www.surfsara.nl`.

[30] The Apache Software Foundation. `http://www.apache.org/`.

[31] The Large Hadron Collider. `http://lhc.web.cern.ch/`.

[32] The LHCb collaboration. `http://lhcb-public.web.cern.ch/lhcb-public/en/Collaboration/Collaboration-en.html`.

[33] The PANDA (Antiproton Annihilation at Darmstadt) Experiment. `http://www.fair-center.eu/public/experiment-program/antiproton-physics/panda.html`.

[34] The Perl Programming Language. `http://www.perl.org/`.

[35] Worldwide LHC Computing Grid (WLCG). `http://lcg.web.cern.ch/lcg/`.

[36] XRootD. `http://xrootd.slac.stanford.edu/`.

[37] Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms. RFC 4513 (Proposed Standard), June 2006.

[38] Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map. RFC 4510 (Proposed Standard), June 2006.

[39] A. Fanfani, A. Afaq, J. A. Sanches et al. Distributed Analysis in CMS. *Journal of Grid Computing*, 8(2):159–179, 2010.

[40] A. Fella, G. Donvito, B. Santeramo et al. SuperB evaluation of DIRAC Distributed Infrastructure. *Journal of Physics: Conference Series*, 396(3):032037, 2012.

[41] A. J. Peters. AliEn Grid User Reference Guide V1.0. `http://project-arda-dev.web.cern.ch/project-arda-dev/alice/apiservice/`.

[42] A. Tsaregorodtsev, V. Garonne, J. Closier et al. DIRAC - Distributed Infrastructure with Remote Agent Control. In *Proceedings of 2003 Conference Computing in High Energy and Nuclear Physics*, (CHEP03), page TUAT006, March 2003. [arXiv:cs/0306060v1].

[43] A. Afaq, A. Dolgert, Y. Guo, C. Jones, S. Kosyakov, V. Kuznetsov, L. Lueking, D. Riley, and V. Sekhri. The CMS dataset bookkeeping service. *Journal of Physics: Conference Series*, 119(7):072001, 2008.

[44] M. Ahsant, J. Basney, O. Mulmo, A. Lee, and L. Johnsson. Toward an On-Demand Restricted Delegation Mechanism for Grids. In *Proceedings of the 7th IEEE/ACM International Conference on Grid Computing*, GRID '06, pages 152–159, 2006.

[45] I. D. Alderman. *A Security Framework for Distributed Batch Computing*. PhD thesis, University of Wisconsin-Madison, April 2010.

[46] I. D. Alderman and M. Livny. Task-specific restricted delegation. In proceedings of the *16th International Symposium on High Performance Distributed Computing*, HPDC '07, 2007.

[47] ALICE Offline Group. Computing Rules. `http://aliweb.cern.ch/Offline/General-Information/ComputingRules.html`.

[48] ALICE Offline Project. AliEn User Howto. `http://alien2.cern.ch/index.php?option=com_content&view=article&id=55&Itemid=95`.

[49] ALICE Offline Project. The ALICE Offline Bible, Version 0.00 (Rev. 22). `http://aliweb.cern.ch/secure/Offline/sites/aliweb.cern.ch.Offline/files/uploads/OfflineBible.pdf`.

[50] S. Bagnasco, L. Betev, P. Bunčić, F. Carminati, C. Cirstoiu, C. Grigoras, A. Hayrapetyan, A. Harutyunyan, A. J. Peter, and P. Saiz. AliEn: ALICE environment on the GRID. *Journal of Physics: Conference Series*, 119(6):062012, 2008.

[51] J. Basney, M. Humphrey, and V. Welch. The MyProxy online credential repository. In *Software: Practice and Experience*, volume 35, pages 801–816, 2005.

[52] K. Benedyczak, P. Bala, S. van den Berghe, R. Menday, and B. Schuller. Key aspects of the UNICORE 6 security model. *Future Generation Computer Systems*, 27(2):195–201, 2011.

[53] D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich, I. Foster, H. Kishimoto, and A. Savva. The Open Grid Services Architecture, Version 1.5. *Open Grid Forum*, July 2006.

[54] K. Bhatia, S. Chandra, and K. Mueller. GAMA: Grid Account Management Architecture. In *e-Science*, pages 413–420. IEEE Computer Society, 2005.

[55] M. Branco, D. Cameron, B. Gaidioz, V. Garonne, B. Koblitz, M. Lassnig, R. Rocha, P. Salgado, and T. Wenaus. Managing ATLAS data on a petabyte-scale with DQ2. *Journal of Physics: Conference Series*, 119(6):062017, 2008.

[56] P. Bunčić, A. J. Peters, and P. Saiz. AliEn Resource Brokers. In *Proceedings of 2003 Conference Computing in High Energy and Nuclear Physics*, number TUAP002 in (CHEP03), March 2003. [arXiv:cs/0306068].

[57] P. Bunčić, A. J. Peters, and P. Saiz. AliEnFS - a Linux File System for the AliEn Grid Services. In *Proceedings of 2003 Conference Computing in High Energy and Nuclear Physics*, number THAT005 in (CHEP03), March 2003. [arXiv:cs/0306071].

[58] P. Bunčić, A. J. Peters, and P. Saiz. The AliEn system, status and perspectives. In *Proceedings of 2003 Conference Computing in High Energy and Nuclear Physics*, number MOAT004 in (CHEP03), March 2003. [arXiv:cs/0306067].

[59] P. Bunčić, C. Aguado Sanchez, J. Blomer, L. Franco, A. Harutyunian, P. Mato, and Y. Yao. CernVM - a virtual software appliance for LHC applications. *Journal of Physics: Conference Series*, 219(4):042003, 2010.

[60] R. Butler, V. Welch, D. Engert, I. Foster, S. Tuecke, J. Volmer, and C. Kesselman. A National-Scale Authentication Infrastructure, December 2000.

[61] A. R. Butt, S. Adabala, N. H Kapadia, R. J. Figueiredo, and J. A. B. Fortes. Grid-computing Portals and Security Issues. *J. Parallel Distrib. Comput.*, 63(10):1006–1014, October 2003.

[62] A. Casajus, K. Ciba, V. Fernandez, R. Graciani, V. Hamar, V. Mendez, S. Poss, M. Sapunov, F. Stagni, A. Tsaregorodtsev, and M. Ubeda. Status of the DIRAC Project. *Journal of Physics: Conference Series*, 396(3):032107, 2012.

[63] A. Casajus and R. Graciani. DIRAC distributed secure framework. *Journal of Physics: Conference Series*, 219(4):042033, 2010.

[64] A. Casajus, R. Graciani, S. Paterson, and A. Tsaregorodtsev. DIRAC pilot framework and the DIRAC Workload Management System. *Journal of Physics: Conference Series*, 219(6):062049, 2010.

[65] A. Ceccanti. A VOMS overview. Technical Report EGEE-II INFSO-RI-031688, NRENS and Grids Workshop, Malaga, November 2007.

[66] A. Chakrabarti, A. Damodaran, and S. Sengupta. Grid Computing Security: A Taxonomy. *IEEE Security and Privacy*, 6(1):44–51, January 2008.

[67] A. P. Chapman, H. V. Jagadish, and P. Ramanan. Efficient provenance storage. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, SIGMOD '08, pages 993–1006. ACM, 2008.

[68] M. Cinquilli, D. Spiga, C. Grandi, J. M. Hernàndez, P. Konstantinov, M. Mascheroni, H. Riahi, and E. Vaandering. CRAB3: Establishing a new generation of services for distributed analysis at CMS. *Journal of Physics: Conference Series*, 396(3):032026, 2012.

[69] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280 (Proposed Standard), May 2008.

[70] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008.

[71] A. Dorigo, P. Elmer, F. Furano, and A. Hanushevsky. XROOTD/TXNetFile: a highly scalable architecture for data access in the ROOT environment. In *Proceedings of the 4th WSEAS International Conference on Telecommunications and Informatics*. World Scientific and Engineering Academy and Society (WSEAS), 2005.

[72] J. Elmsheuser and D. van der Ster. Distributed Data Analysis in the ATLAS Experiment: Challenges and Solutions. *Journal of Physics: Conference Series*, 396(3):032035, 2012.

[73] F. Farina, S. Lacaprara, W. Bacchi et al. Status and evolution of CRAB. *Proceedings of Science (PoS)*, XI International Workshop on Advanced Computing and Analysis Techniques in Physics Research(ACAT2007:020), 2007.

[74] D. Feichtinger and A. J. Peters. Authorization of Data Access in Distributed Storage Systems. In *The 6th IEEE/ACM International Workshop on Grid Computing*, Grid'05, 2005.

[75] I. Foster. What is the Grid? - a three point checklist. *GRIDtoday*, 1(6), July 2002.

[76] I. Foster and C. Kesselman. Globus: A Toolkit-Based Grid Architecture. In I. Foster and C. Kesselman, editors, *The Grid: Blueprint for a New Computing Infrastructure*, pages 259–278. Morgan Kaufmann, 1999.

[77] I. Foster and C. Kesselman. Concepts and Architecture. In I. Foster and C. Kesselman, editors, *The Grid: Blueprint for a New Computing Infrastructure*, pages 37–63. Morgan Kaufmann, 2nd ed., 2004.

[78] I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke. A Security Architecture for Computational Grids. In *Proceedings of the 5th ACM conference on Computer and communications security*, CCS '98, pages 83–92, 1998.

[79] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of High Performance Computing Applications*, 15(3), August 2001.

[80] I. Foster, J. Vöckler, M. Wilde, and Y. Zhao. Chimera: A Virtual Data System For Representing, Querying, and Automating Data Derivation. In *In Proceedings of the 14th Conference on Scientific and Statistical Database Management*, pages 37–46, 2002.

[81] G. A. Stewart, V. Garonne, M. Lassnig et al. Advances in service and operations for ATLAS data management. *Journal of Physics: Conference Series*, 368(1):012005, 2012.

[82] Geo Grid. Global Earth Observation Grid. `http://www.geogrid.org/en/index.html`.

[83] A. Gheata. ALICE Analysis Framework. *Proceedings of Science (PoS)*, XII International Workshop on Advanced Computing and Analysis Techniques in Physics Research(ACAT08:028), 2008.

[84] J. Greene. Intel Trusted Execution Technology, Intel Corporation, White Paper. `http://www.intel.com/content/dam/www/public/us/en/documents/white-papers/trusted-execution-technology-security-paper.pdf`.

[85] D. Groep. glexec deployment models - local credentials and grid identity mapping in the presence of complex schedulers. Technical Report INFSO-RI-508833, Joint OSG EGEE Operations Workshop CERN, Geneva, June 2006.

[86] D. Groep, O. Koeroo, and G. Venekamp. gLExec: gluing grid computing to the Unix world. *Journal of Physics: Conference Series*, 119(6):062032, 2008.

[87] A. Harutyunyan, J. Blomer, P. Bunčić, I. Charalampidis, F. Grey, A. Karneyeu, D. Larsen, D. Lombraña González, J. Lisec, B. Segal, and P. Skands. CernVM Co-Pilot: an Extensible Framework for Building Scalable Computing Infrastructures on the Cloud. *Journal of Physics: Conference Series*, 396(3):032054, 2012.

[88] A. Harutyunyan, P. Bunčić, T. Freeman, and K. Keahey. Dynamic virtual AliEn Grid sites on Nimbus with CernVM. *Journal of Physics: Conference Series*, 219(7):072036, 2010.

[89] S. Jézéquel and G. Stewart. ATLAS Distributed Computing Operations: Experience and improvements after 2 full years of data-taking. *Journal of Physics: Conference Series*, 396(3):032058, 2012.

[90] Joint Security Policy Group. Grid Security Policy. Technical Report CERN-EDMS-428008, LCG EGEE, 2007.

[91] Joint Security Policy Group. Grid Security Traceability and Logging Policy. Technical Report CERN-EDMS-428037, LCG EGEE, 2008.

[92] Joint Security Policy Group. Grid Acceptable Use Policy. Technical Report CERN-EDMS-428036, LCG EGEE, 2010.

[93] D. Kouril and J. Basney. A Credential Renewal Service for Long-Running Jobs. In *Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing*, GRID '05, pages 63–68, 2005.

[94] C. Lefevre. The LHC Guide, CERN-Brochure-2009-003, February 2009. `http://cds.cern.ch/record/1165534/files/CERN-Brochure-2009-003-Eng.pdf`.

[95] LHCb Computing Group. Workload Management with Pilot Agents in DIRAC - LHCb Technical Note. Technical Report LHCB Comp 08-006, LHCb, CERN, 2008.

[96] M. Litmaath. Authentication and Authorization Infrastructure for Data Storage and Access. *WLCG Technical Report*, April 2012.

[97] M. Aderholz, K. Amako, E. Auge et al. Models of Networked Analysis at Regional Centres for LHC Experiments, (MONARC), PHASE 2 REPORT. Technical Report CERN-LCB-2000-001. KEK-2000-8, CERN, Geneva, April 2000.

[98] T. Maeno. PanDA: Distributed Production and Distributed Analysis System for ATLAS. *Journal of Physics: Conference Series*, 119(6):062036, 2008.

[99] T. Maeno, K. De, and S. Panitkin. PD2P: PanDA Dynamic Data Placement for ATLAS. *Journal of Physics: Conference Series*, 396(3):032070, 2012.

[100] M. Mambelli. The ATLAS Experiment on the Grid, Mid-West Grid School (MWGS'08), September 2008. `https://www.opensciencegrid.org/twiki/bin/viewfile/Education/MWGS2008Syllabus/atlas.pdf`.

[101] B. C. Neuman. Proxy-Based Authorization and Accounting for Distributed Systems. In proceedings of the *13th International Conference on Distributed Computing Systems*, pages 283–291, 1993.

[102] H. F. Nielsen, N. Mendelsohn, J. I. Moreau, M. Gudgin, M. Hadley, A. Karmarkar, and Y. Lafon. SOAP Version 1.2 Part 1: Messaging Framework (Second Edition). W3C Recommendation, W3C, April 2007.

[103] P. Nilsson, J. Caballero, K. De, T. Maeno, M. Potekhin, and T. Wenaus. The PanDA System in the ATLAS Experiment. *Proceedings of Science (PoS)*, XII International Workshop on Advanced Computing and Analysis Techniques in Physics Research(ACAT08:027), 2008.

[104] OASIS Technical Committee. OASIS eXtensible Access Control Markup Language (XACML) v3.0, 2013. `http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.pdf`.

[105] OASIS Technical Committee. OASIS Security Assertion Markup Language (SAML) v2.0, 2005. `http://docs.oasis-open.org/security/saml/v2.0/sstc-saml-approved-errata-2.0.pdf`.

[106] Oracle Corporation. Java Platform Standard Ed. 7 - Class KeyStore. `http://docs.oracle.com/javase/7/docs/api/java/security/KeyStore.html`.

[107] Oracle Corporation. Java Remote Method Invocation - Distributed Computing for Java. `http://www.oracle.com/technetwork/java/javase/tech/index-jsp-138781.html`.

[108] OSG security team. Open Science Grid User Acceptable Use Policy. Technical Report OSG Document 86, OSG, 2006.

[109] PANDA Collaboration. The Computing Model of PANDA. Technical report, FAIR, April 2009.

[110] S. K. Paterson. LHCb gLExec Testing. In *CERN GDB Meeting Geneva*, November 2008.

[111] P. Saiz, L. Aphecetche, P. Bunčić, R. Piskač, J.-E. Revsbech, V. Šego, and ALICE Collaboration. AliEn-ALICE environment on the GRID. *Nuclear Instruments and Methods in Physics Research A*, 502:437–440, 2003.

[112] M. Schmidt, M. Smith, N. Fallenbeck, H. Picht, and B. Freisleben. Building a demilitarized zone with data encryption for grid environments. In *In Proceedings of First International Conference on Networks for Grid Applications*, pages 8–16. IEEE Press, 2007.

[113] K. Schwarz, F. Uhlig, R. Karabowicz, A. Montiel-Gonzalez, M. Zynovyev, and C. Preuss. Grid Computing at GSI for ALICE and FAIR - present and future. *Journal of Physics: Conference Series*, 396(3):032097, 2012.

[114] Security Policy Group. EGI-InSPIRE Grid Acceptable Use Policy. Technical Report EGI-SPG-AUP-V1_0, EGI, 2010.

[115] Security Policy Group. EGI-InSPIRE Grid Security Policy. Technical Report EGI-SPG-SecurityPolicy-V1_0, EGI, 2010.

[116] Security Policy Group. EGI-InSPIRE Grid Security Traceability and Logging Policy. Technical Report EGI-SPG-Traceability-V1_0, EGI, 2010.

[117] S. Sekiguchi, Y. Tanaka, I. Kojima, N. Yamamoto, S. Yokoyama, Y. Tanimura, R. Nakamura, K. Iwao, and S. Tsuchida. Design Principles and IT Overviews of the GEO Grid. *IEEE Systems Journal*, 2(3):374–389, 2008.

[118] I. Sfiligoi. glideinWMS - a generic pilot-based workload management system. *Journal of Physics: Conference Series*, 119(6):062044, 2008.

[119] S. Shahand, M. Santcroos, A. H. van Kampen, and S. Delgado Olabarriaga. A Grid-Enabled Gateway for Biomedical Data Analysis. *Journal of Grid Computing*, 10:725–742, 2012.

[120] A. C. Smith and A. Tsaregorodtsev. DIRAC: reliable data management for LHCb. *Journal of Physics: Conference Series*, 119(6):062045, 2008.

[121] M. Smith, M. Engel, T. Friese, B. Freisleben, G. A. Koenig, and W. Yurcik. Security issues in on-demand grid and cluster computing. In *In CCGRID '06: Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, volume 2. IEEE Computer Society, 2006.

[122] M. Smith, T. Friese, M. Engel, and B. Freisleben. Countering Security Threats in Service-oriented On-demand Grid Computing Using Sandboxing and Trusted Computing Techniques. *J. Parallel Distrib. Comput.*, 66(9):1189–1204, September 2006.

[123] M. Smith, M. Schmidt, N. Fallenbeck, T. Dörnemann, C. Schridde, and B. Freisleben. Secure On-demand Grid Computing. *Future Gener. Comput. Syst.*, 25(3):315–325, March 2009.

[124] D. F. Snelling, S. van den Berghe, and V. Q. Li. Explicit trust delegation: Security for dynamic grids. *Fujitsu Scientific and Technical Journal*, 40(2):282–294, 2004.

[125] T. M. Kurç, S. Hastings, V. S. Kumar et al. HPC and Grid Computing for Integrative Biomedical Research. *Int. Journal of High Performance Computing Applications*, 23:252–264, 2009.

[126] The ALICE Computing Group. ALICE computing technical design report. Technical Report CERN-LHCC-2005-018, ALICE-TDR-012, CERN, 2005.

[127] The ATLAS Computing Group. ATLAS computing technical design report. Technical Report CERN-LHCC-2005-022, ATLAS-TDR-017, CERN, 2005.

[128] The CMS Computing Group. CMS computing technical design report. Technical Report CERN-LHCC-2005-023, CMS-TDR-007, CERN, 2005.

[129] The LCG TDR Editorial Board. LHC computing Grid technical design report. Technical Report CERN-LHCC-2005-024 ; LCG-TDR-001, CERN, 2005.

[130] The Legion of the Bouncy Castle. Bouncy Castle API. `http://www.bouncycastle.org/`.

[131] The LHCb Computing Group. LHCb computing technical design report. Technical Report CERN-LHCC-2005-019, LHCb-TDR-011, CERN, 2005.

[132] S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson. Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. RFC 3820 (Proposed Standard), June 2004.

[133] V. Welch, I. Foster, C. Kesselman, O. Mulmo, L. Pearlman, J. Gawor, S. Meder, and F. Siebenlist. X.509 proxy certificates for dynamic delegation. In *Proceedings of the 3rd Annual PKI R&D Workshop*, 2004.

[134] T. Wenaus. The ATLAS Panda Production and Distributed Analysis System, Technology Meeting BNL, March 2006. `https://www.racf.bnl.gov/Facility/TechnologyMeeting/Archive/Mar-13-2006/200602-panda-tech.pdf.`

[135] J. Wu, R. Siewert, A. Hoheisel, J. Falkner, O. Strauß, D. Berberovic, and D. Krefting. The Charité Grid Portal: User-friendly and Secure Access to Grid-based Resources and Services. *Journal of Grid Computing*, 10(4):709–724, 2012.

Wissenschaftlicher Werdegang

Mai 2009 - Dezember 2014 Promotionsstudent bei Prof. Dr. Johannes Buchmann am Lehrstuhl für Kryptographie und Computeralgebra an der Technischen Universität Darmstadt

Mai 2009 - April 2012 Wissenschaftlicher Mitarbeiter im ALICE Experiment der European Organization for Nuclear Research in Genf in einer Kooperation mit der Technischen Universität Darmstadt und dem Center for Advanced Security Research Darmstadt unter der Betreuung von Prof. Dr. Johannes Buchmann

Oktober 2001 - April 2009 Studium der Informatik (Diplom) an der Technischen Universität Darmstadt

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit - abgesehen von den in ihr ausdrücklich genannten Hilfen - selbständig verfasst habe.

Darmstadt, November 2014 ____________________________