Access Control and User Mapping at a WLCG Tier-2 Site

Oscar Kraemer

Bachelor
Information Technology
June 1, 2015

BACHELOR	
Arcada University of Applied Sciences	
Degree Programme:	Information Technology
Identification number:	13179
Author:	Oscar Kraemer
Title:	Access Control and User Mapping at a WLCG Tier-2 Site
Supervisor (Arcada):	Göran Pulkkis
Commissioned by:	Helsinki Institute of Physics

Abstract:
The goal of this thesis is to create a working system for access control and user mapping of pilot jobs for the computing resources at Helsinki Institute of Physics. The thesis work shows how to create a centralized system for access control and user mapping with the software components ARGUS, GLExec, and ARC. The thesis work was done to fulfill Worldwide LHC Computing Grid's (WLCG) new requirements that all WLCG tier sites must use GLExec. The thesis work presents the software components needed, describes how they are used, and how access control and user mapping works at Helsinki Institute of Physics.

Keywords:	CMS, Grid, Argus, ARC, GLExec, PepCli, Certification, User Mapping, Grid Mapping
Number of pages:	67
Language:	English
Date of acceptance:	28.05.2015

INGENGÖR	
Yrkeshögskolan Arcada	
Utbildningsprogram:	Informations teknik
Identifikationsnummer:	13179
Författare:	Oscar Kraemer
Arbetets namn:	Åtkomstkontroll och användarregistrering på en WLCG Tier-2 site
Handledare (Arcada):	Göran Pulkkis
Uppdragsgivare:	Forskningsinstitutet för fysik

Sammandrag:
Målet med detta examensarbete är att skapa ett fungerande system för åtkomstkontroll och användarregistrering av pilotberäkningsuppgifter för beräkningsresurserna vid Forskningsinstitutet för fysik. Slutarbetet visar hur man kan skapa ett centraliserat system för åtkomstkontroll och användarregistrering med mjukvarukomponenterna ARGUS, GLExec och ARC. Arbetet gjordes för att uppfylla WLCGs (Worldwide LHC Computing Grid) nya krav på att alla WLCG Tier sites skall använda GLExec. Examensarbetet presenterar de nödvändiga mjukvarukomponenterna, beskriver hur de utnyttjas samt hur åtkomstkontrollen och användarregistreringen fungerar vid Forskningsinstitutet för fysik.

Nyckelord:	CMS, Grid, Argus, ARC, GLExec, PepCli, Certification, User Mapping, Grid Mapping
Sidantal:	67
Språk:	Engelska
Datum för godkännande:	28.05.2015

CONTENTS

FIGURES

ABBREVIATIONS

ALICE	A Large Ion Collider Experiment
ARC	Advanced Resource Connector
CA	Certification Authority
CERN	European Organization for Nuclear Research
CLI	Command Line Interface
CMS	Compact Muon Solenoid
FQAN	Fully Qualified Attribute Name
GID	Group Identifier
HIP	Helsinki Institute of Physics
KVM	Kernel-based Virtual Machine
M-grid	Finnish Material Sciences Grid
MUNGE	MUNGE Uid 'N' Gid Emporium
NDGF	Nordic Data Grid Facility
NIS	Network Information Service
NFS	Network File System
NTP	Network Time Protocol

LHC	The Large Hadron Collider
LHCb	Large Hadron Collider beauty experiment
OS	Operating System
PAP	Policy Administration Point
PDP	Policy Decision Point
PEP	Policy Execution Point
PEPCLI	Policy Execution Point Command Line Interface
PKI	Public Key Infrastructure
SL	Scientific Linux
SL6	Scientific Linux version 6
SLURM	Simple Linux Utility for Resource Management
UH	University of Helsinki
UID	User Identifier
VM	Virtual Machine
VO	Virtual Organization
VOMS	Virtual Organization Membership Service
WN	Worker Node

WLCG	Worldwide LHC Computing Grid
YAIM	YAIM Ain't an Installation Manager
YUM	Yellowdog Updater, Modified

1 INTRODUCTION

Because computers do not always work correctly a new type of grid jobs, pilot jobs, has been introduced. A pilot job is sent to a cluster to execute a test if the cluster is working correctly. If the cluster is working correctly, the pilot job pulls in the real job that will do calculations. In this type of setup a new problem occurs when the cluster owner can't know who owns the real jobs. This is a security concern. To be able to solve this problem we have to check who is the owner of the real job and then map the job to the real owner. This thesis work shows how it can be done, with the software components ARC, GLExec, and ARGUS.

1.1 Background

The European Council for Nuclear Research (CERN) founded in 1954 has as a mission to provide the instruments to study the fundamental particles of the universe. To study these small particles CERN needs to build custom instruments, accelerators and detectors. CERN is located on the French-Swiss border (About CERN, 2013).

The Large Hadron Collider (LHC) is the world's largest particle accelerator and the latest accelerator built by CERN. LHC was started September 10th 2008. While being the largest accelerator in the world it is also the most powerful. The collider consists of a tube that is formed in a 27 km long circle where particles are accelerated by electric fields. When particles have reached enough speed they are made to collide into each other. The inside of the accelerator is kept in a vacuum and the temperature is close to absolute zero. The particles are traveling close to the speed of light when they are collided into each other (The Large Hadron Collider, 2014). There are four large LHC experiments (ATLAS, CMS, ALICE and LHCb) with their own detectors located on four places around the accelerator where collisions are possible. The Compact Muon Solenoid

(CMS) is one of the four large LHC detectors (CMS, 2014). The detector weight is about 12 000 tones and it has a diameter of 15 meters (Why is CMS so big?, 2011).

1.2 WLCG

The Worldwide LHC Computing Grid (WLCG) is the organisation that provides the infrastructure and computing resources which are needed for research on the data LHC generates. To be able to use the WLCG resources you have to be part of a Virtual Organisation (VO) like the CMS Experiment. The detectors produce a lot of data, the CMS detector alone produces more than 5 petabytes/year. To be able to process all this data the CMS experiment uses the Worldwide LHC Computing Grid (WLCG) to distribute the data and the computing across the world (Computing Grid, 2011).

1.3 CMS Computing Model

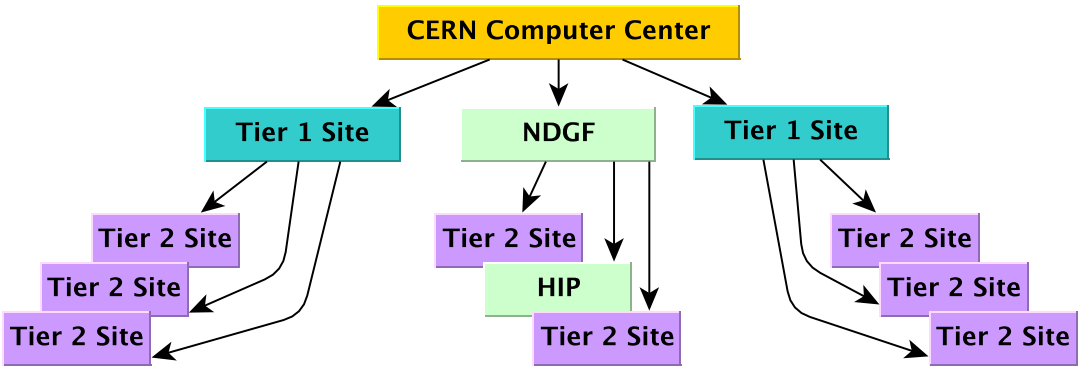

Figure 1. CMSCM site hierarchy

In CMS Computing Model (CMSCM) the Tier sites are organized in a three level hierarchical structure, a simple visualization is shown in Figure 1.A Tier site consists of different kinds of computing and storage resources. The main Tier-0 is based at CERN where the data is reconstructed from the CMS detector. This is basically the point that

needs to be able to pass through 5 petabyte per year. The Tier-0 then passes the data to the Tier-1 sites, where the data is archived, reconstructed and other heavy data intensive jobs are executed. Tier-2 sites are used for data intensive and CPU intensive processes. There are more Tier-2 sites than Tier-1 sites and they use mostly the Tier-1 sites to get physics data. The Tier-2 sites are relied on when it comes to analysis and Monte Carlo simulations. Tier-3 sites are the smallest and used by local institutes and can be used when possible as grid resources. (The CMS Computing Project Technical Design Report, 2005)

1.3.1 Clusters

In this thesis work the focus is on clusters. A cluster is built up by many interconnected computers (servers) called nodes to make it from the outside seem like a single computer. Supercomputers and mainframes are very good at doing calculations very fast but are very expensive. Clusters are built up by normal off-the-shelf servers and have the benefit that they are less expensive. A cluster might not be good to do heavy calculations but is good at doing many tasks at the same time for less money than a mainframe or a supercomputer. A cluster is also a good alternative to an arrangement where all researchers have their own very powerful workstations that are used to do calculation when needed. Faculties, universities and other groups can pool their money and buy a joint cluster with the assumption that everybody doesn't need to use the computing resources at the same time and in this way be able to get a higher resource utilisation and save money.

1.3.2 Pilot Jobs

Clusters are not always working perfectly and often something is broken, it can be anything from a broken hard drive to a bad update. Even if there are tests that are running on the clusters to check if the clusters are broken, there might still be problems going un-

noticed. The users hate when their jobs have first queued and then fail because a cluster node has become faulty. To solve the problem of jobs failing because a test has not noticed a bad node or bad configurations, there are pilot factories that are sending out pilot jobs. These jobs queue into clusters and when they get permission to run they check if the resources are working correctly. If a resource is not working correctly the pilot job fails. If a resource works properly the pilot job will pull in a job that will do the actual calculations. A problem that arises by the pilot job solution, is that when a pilot job accesses a resource authorizied by a certificate, but the actual job is submitted by another user and has an another certificate. To be able to know who is actually running on our clusters we need to be able to check who the real user is and give this user a valid permission. This will prevent pilot jobs to be used as Trojans. This thesis report presents an implemented solution.

1.4 Helsinki Institute of Physics

Helsinki Institute of Physics (HIP), founded in 1996, is a joint institute of University of Helsinki (UH), University of Jyväskylä, Lappeenranta University of Technology, and Tampere University of Technology. HIP is physically located in UH facilities but there is personnel also in the other universities and at CERN (Annual Report 2013 Helsinki Institute of Physics, 2014) . HIP studies particle physics and has a variety of programmes, one of them is the CMS Programme. In this programme there is the CMS Tier-2 Operations Project for which this thesis report is written.

1.4.1 HIP Tier-2 Site

HIP has a CMS Tier-2 site that consists of three Linux clusters, Korundi, Jade and Alcyone. Korundi, the oldest cluster, bought in 2008, fitted with 400 cores, and located in Kumpula, Helsinki is used not only by CMS but also by the department of Physics and

Chemistry of UH. Jade was bought in 2009, fitted with 768 cores, and located in Espoo at the Finnish IT Center for Science (CSC). The newest Cluster Alcyone, bought in 2011, has 840 cores, and is used by CMS (Annual Report 2013 Helsinki Institute of Physics, 2014) and of Physics, Chemistry and Computer Science departments at UH (Alcyone Cluster, 2013).

1.5 Goals

The WLCG management board has decided that all sites shall use GLExec (gLExec Deployment Tracking, 2015), which needs to be implemented. All UH clusters are using Simple Linux Utility for Resource Management (SLURM) (Overview, 2013) software which won't be changed so it won't be inconvenient for other users. The Advanced Resource Connector (ARC) (About Advanced Resource Connector, 2015) software is already being used on the clusters so it won't be changed. Since major software components are already chosen, the task of this thesis work is to make the components work together. Another goal is to cause as little inconvenience as possible for the other users of the clusters. A decision to use Argus software (Argus System Administrator Guide, 2013) has been made.

2 CERTIFICATION

To know who is who on the Internet the use of PKI certificates is needed. Certificates are used to verify the identity of computers and persons. A certificate is much like a government issued passport. We trust that a country only issues passports to persons that the government have verified. We can't trust a passport if we can't trust that the government only issues valid passports. We also need to verify that a passport is valid, has not expired, and is not a fake. Also for electronic certificates we have to be sure that they are valid. Certificates can be revoked if there is a reason for it. (What are

Certificates?, 2013).

There is no reason to send data encrypted if we can't be sure of the identity of the receiver. In this thesis report it is assumed that the reader is familiar with Public Key Infrastructure (PKI) and certification (Background Reference: What are Certificates and PKI?, 2015). CERN has its own Certification Authority (CA) that is used to authorize different server services which use the grid as well as personal services like e-mail, web applications, and workstations (Ormancey, 2014).

2.1 Proxy Certificate

Many grid jobs need access to services that require a user certificates for validation. One way would be to send with the job the individual user's certificate. This would however be a great security risk for the whole grid, because then someone would easily be able to steal the certificate and do some serious damage. To be able to avoid this problem when a user needs to send a job with some inherent permissions, the user will create a proxy. When creating a proxy a new private-public key pair is created and the new proxy certificate is signed with the user's normal private key. This is an advantage because then the user's private key is not in risk to be stolen. An advantage with a proxy certificate is also that it has a short lifetime, usually 12 hours from creation (USG Proxy Certificates, 2013). This means that if the proxy gets highjacked then a bad guy has at most the proxy lifetime available to do damage before the proxy certificate expires. A drawback might be that if there is a long queue to a computing resource, then the proxy certificate might expire before the job has had a chance to be accepted.

2.1.1 Virtual Organization Membership Service

Virtual Organization Membership Service (VOMS) is a centralised "repository" that provides an attribute based authorization for the grid users. For a grid users credentials to be served by a VOMS the user need to be part of a Virtual Organization (VO). In WLCG the VOs are the same as the main research groups (the experiments); CMS, ALICE, ATLAS, etc. (Virtual Organisations, 2015).

So for a user to be able to submit grid jobs, the user must have a certificate issued by a trusted CA and the user must also be a member of one of the WLCG's VOs (Setting Up KPI : Certificates and VOMS, 2015).

The services that are VOMS aware verify grid users by the proxy certificate that is signed by a trusted VOMS service (How to configure VOMS LSC files, 2014). Examples of services that are VOMS aware are ARC and Argus. In our setup only Argus' VOMS capabilities are used.

When a user wants to send a grid job, a proxy certificate is needed and the user must have VOMS attributes that are accepted by the resource where he wants to send his job. For example HIP is a part of CMS and wants to accept cms-VO but does not want to provide free computing resources to the ATLAS resource group. The user will initialize a proxy certificate with the command line tool voms-proxy-int (voms-proxy-init, 1) or the nordugrid tool arcproxy (arcproxy, 1) where he specifies a VOMS attribute. Example VOMS attributes can be found in APPENDIX 9.

2.1.2 Creating a Proxy Certificate

As I am a member of the cms group and have an accepted certificate I can create a proxy certificate like this:

```
[oscar@alcyone ~]$ voms-proxy-init --voms cms
Enter GRID pass phrase:
Your identity: /DC=org/DC=terena/DC=tcs/C=FI/O=University of Helsinki/
CN=CarlKraemer cokraeme@helsinki.fi
Creating temporary proxy ..........................................
..... Done
Contacting  lcg-voms2.cern.ch:15002 [/DC=ch/DC=cern/OU=computers/CN=lc
g-voms2.cern.ch] "cms" Done
Creating proxy ....................................................
.....................................................................
..... Done
```

I only need to specify "–voms=cms" because the necessary values to connect to the VOMS server can be found in /etc/vomses. My certificate is in the standard directory " /.globus/". With the voms-proxy-info command you can find out more about the proxy. You can see the subject line contains a "CN=proxy", and more information about the attributes. You can also se that I do not have any special attributes other than the "cms" and therefore the Role and Capability are "=NULL". Also note the standard "timeleft" that is a couple of seconds below 12 hours.

```
[oscar@alcyone ~]$ voms-proxy-info -all
subject   : /DC=org/DC=terena/DC=tcs/C=FI/O=University of Helsinki/
CN=Carl Kraemer cokraeme@helsinki.fi/CN=proxy
issuer    : /DC=org/DC=terena/DC=tcs/C=FI/O=University of Helsinki/
CN=Carl Kraemer cokraeme@helsinki.fi
identity  : /DC=org/DC=terena/DC=tcs/C=FI/O=University of Helsinki/
CN=Carl Kraemer cokraeme@helsinki.fi
type      : proxy
```

```
strength  : 1024 bits
path      : /tmp/x509up_u757
timeleft  : 11:59:52
key usage : Digital Signature, Key Encipherment
=== VO cms extension information ===
VO        : cms
subject   : /DC=org/DC=terena/DC=tcs/C=FI/O=University of Helsinki/
CN=Carl Kraemer cokraeme@helsinki.fi
issuer    : /DC=ch/DC=cern/OU=computers/CN=lcg-voms2.cern.ch
attribute : /cms/Role=NULL/Capability=NULL
timeleft  : 11:59:52
uri       : lcg-voms2.cern.ch:15002
```

Fully Qualified Attribute Names (FQAN) in a proxy certificate can be used to describe all the attributes of a proxy certificate. (Proxy Certificates, 2008)

```
[oscar@alcyone .globus]$ voms-proxy-info -fqan
/cms/Role=NULL/Capability=NULL
```

3 SOFTWARE COMPONENTS

To be able to create our access and mapping system we must combine some software components, see Figure 2. In this chapter we present the required software components.

3.1 ARC

The Advanced Resource Connector (ARC) is located on a cluster's frontend nodes. ARC is a grid middleware, which is the software that connects the cluster to the grid. The grid

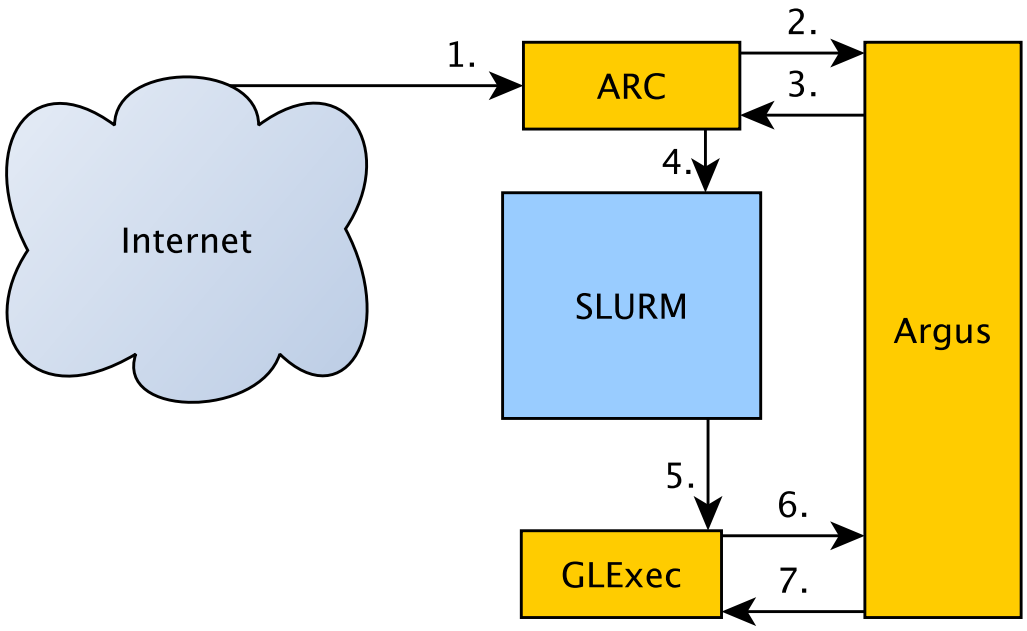

Figure 2. The software components' relation to each other

middleware is basically the door between the cluster and the grid (The Grid: Software, middleware, hardware, 2015). Most European WLCG resources are using the gLite middleware (Introduction, 2015). ARC is mostly used in the Nordic countries: SweGrid, DCGC, and NDGF (Koivumäk, 2010).

3.1.1 Nordugrid

The NorduGrid Collaboration (2015) was founded in 2001 by the Nordic countries. To be able to have a working grid one needs some kind of software to bundle the computing resources together, or else it is just a bunch of computing resources on the Internet. In NorduGrid the software is ARC. Even if a resource is connected to a grid by ARC, the resource owner still has full control of the resource and can decide who has access to the resource. P.Eerola (2003)

3.2 Argus

The Argus Authorization Service is used as an authorization decision service and a user mapping service. The decisions are based on grid jobs's proxy certificates. On our grid site, HIP CMS Tier-2, there are multiple services that need consistent authorization and user mapping. ARC must know if a grid job has authorization to the cluster and also to what UNIX user the job should map to. Also gLExec gLExec-wn (2015) must know to what UNIX user a job shall map to. Argus is capable of much more advanced use than this thesis report describes. We will only use it to permit and deny access and to map a certificate to a specific UNIX user. The benefit of using Argus is that we only need to create access policies for all of our sites once and then all our clusters and services can use Argus to make their authorization decisions.

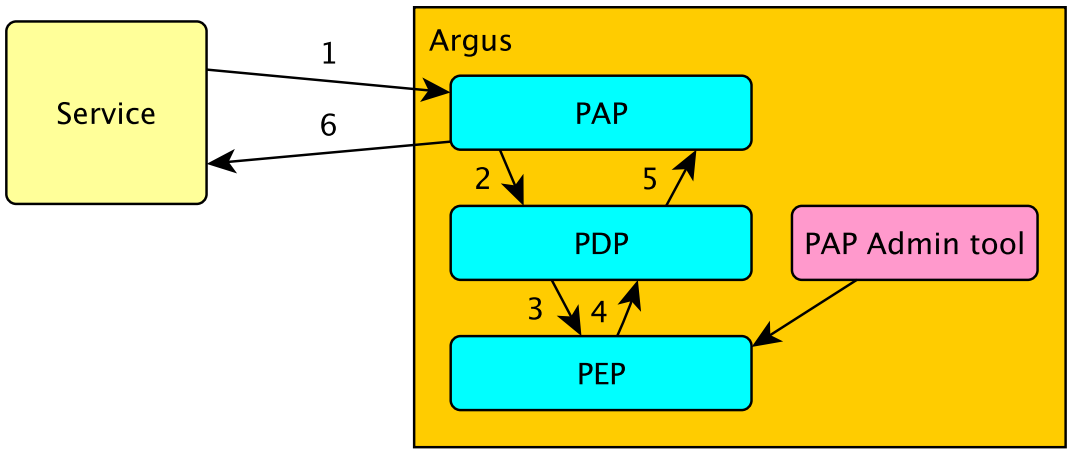

Figure 3. Argus software components' relation to each other

Argus consists of three components, Policy Execution Point (PEP), Policy Decision Point (PDP), and Policy Administration Point (PAP), see Figure 3. PEP is the component that we use when we request a policy decision over a network. PEP takes the request to PDP that makes a decision on the policies that have been inputed to PAP. We will use thePAP Admin tool to create our policies.

3.3 PEPCLI

PEPCLI is an application that is used to to send requests to Argus. We can use PEPCLI as a command to check if a proxy certificate has permission to run on our clusters and to what UNIX name the owner of the proxy certificate should map to. By using PEPCLI we can get responses like "Permit", "Deny", "Applicable" and "Indeterminate". If Argus is able to map the proxy certificate to a UNIX user we can get the UNIX name and group name as a response.

3.3.1 Wrapper Script

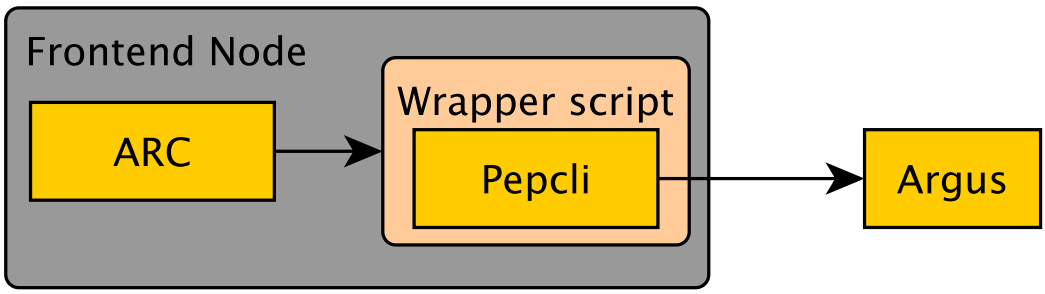

Figure 4. The Wrapper script's part of the communication

For ARC to be able to contact Argus we need to use PEPCLI, but because ARC can't use PEPCLI directly we need to use PEPCLI inside a wrapper script, see Figure 4.

The wrapper script in Appendix 1 takes as input (sys.argv[1]) the location of the proxy certificate that one wants to check. Then the script calls PEPCLI (/usr/bin/pepcli) with the necessary parameters. PEPCLI returns the response to the variable "input" which is thereafter parsed to a format understandable to ARC. The parsed output is: R:* , where R is a local UNIX user name.

In an earlier version the wrapper script also returned the UNIX group name. However,

we realised that ARC would not need it as ARC would map a user to its primary group in either case. This is a good thing for us since it makes the problem area smaller. The * in R:* represents any group name. If Argus manages to respond with a UNIX name, then the wrapper script will output the R:* to ARC and exit with the sucsessful return value 0 (sys.exit(0)). ARC therefore knows that the proxy certificate's owner has permission to use the cluster. If Argus fails to map the user, then the wrapper script will execute the voms-proxy-info command with the proxy certificate's location as a variable to find out as much information as possible about the proxy certificate and write this information combined with the time into the log file /var/log/oscars-messages. Then the script will exit with the return value 1 so that ARC knows that Argus was not able to "Permit" the owner to use the cluster.

3.4 GLExec

GLExec is an application that is installed on all worker nodes. GLExec switches the UNIX mapping of a pilot job to the mapping of the user of the job that the pilot job pulled in.

When a pilot job has passed ARC and been submitted by SLURM Overview (2013) to a worker node, it runs an environment check script that basically checks, if the environment on the worker node contains all libraries, paths, a.s.o. that are necessary for a grid job to be executed correctly on the worker node. If the pilot job is successful without finding any errors (the environment is correct) then the pilot job pulls in a grid job that without GLExec would have the same mapping as the pilot job. GLExec is now called to switch user for the grid job. GLExec should only be permitted to run by UNIX users that are authorized (white listed) to run pilot jobs. An example can be seen in APPENDIX 3. GLExec will use the grid job's certificate as an input variable and then send a request to Argus-PEP to get the correct UNIX user mapping, that GLExec will configure the grid

job to run under.

3.5 NIS

The Network Information Service (NIS) (Homepage of the Linux NIS/NIS+ Projects, 2012) is used on the clusters to keep UNIX mapping consistent over all nodes on a cluster. It is essential for the user mapping that the UNIX user name and the UNIX id are consistent on the cluster, not only for the user's spaces that are shared over the clusters by NFS (Network File System) but also for Argus to be able to make decisions that are correct on the clusters. Every cluster has its own working NIS from the installation of the cluster.

3.6 SLURM

The Simple Linux Utility for Resource Management (SLURM) (Overview, 2013) is a job scheduler that is used on many clusters and supercomputers. A job schedulers task is to distribute jobs on a cluster. In this thesis work ARC is submitting jobs to the SLURM queue and SLURM will make decisions on which job will run on which node and when. SLURM is configured to give higher priority to some jobs based on the owner of the jobs. SLURM knows the owner of the job because ARC has already given a UNIX name to the job as is described earlier.

3.6.1 Munge

MUNGE Uid 'N' Gid Emporium (Munge) (munge, 2015) is installed on all nodes on the clusters and is used as a secure way to keep user ids and group ids consistent between the nodes inside SLURM.

4 INFRASTRUCTURE

In this chapter we present the hardware we are using to get a better understanding of the environment in use.

4.1 Virtualization

Nowadays computers are powerful and if you have relatively lightweight applications, then it might be a good thing to let these applications run on a virtualized machine (VM). You can save cost when you only need one physical machine running multiple virtual machines. The physical server that hosts a VM is often referred to as a host. Argus will be running on a VM. Benefits with virtualization is that a new VM can rapidly be deployed which gives higher flexibility than for physical machines. I will create two identical instances of Argus on two different hosts. One instance is actively running, one is powered down, and can be deployed swiftly, if the first VM encounters an unrecoverable problem and goes down or if the host has a malfunction. I will use Kernel-based Virtual Machine (KVM) (Kernel Based Virtual Machine, 2015) for virtualization and virt-manager (Manage virtual machines with virt-manager, 2013) to manage the virtual machines.

4.2 Operating System

The Operating System (OS) Scientific Linux (SL) (About, 2014) version 6 (SL6) is being used, as it was the newest version of SL when this thesis work was done. SL is a Linux/GNU distribution based on CentOS with the same version number. CentOS is in its turn based on Red Hat Enterprise Linux that is based on Fedora. SL is created in collaboration with Fermilab and CERN (Why make Scientific Linux?, 2014) and is the natural choice of distribution for CERN related computer services.

4.3 Hardware

All hardware in use is server grade with ECC memory and Xeon processors. Most servers have at least 8 or more CPU cores with a few exceptions. The total amount of cores in this thesis work is about 2000 in about 200 servers.

4.3.1 Carbon and Phedex

Carbon and Phedex are two older servers that we are using as virtualization hosts. Both have enough memory and CPU power to be able to run Argus as well as some other VMs.

4.4 Cluster Nodes

A worker node (WN) is a physical server in a cluster. A WN is a machine that is doing the calculations. In the cluster there are also frontend nodes. Frontend nodes are nodes that are connecting the cluster to the internet. Alcyone, Korundi and Jade have a frontend node that local users can log in to, a VM for every different grid the cluster is connected to, and a physical admin node that is functioning as a VM host and as a NFS node. On Korundi the NFS node and login node is the same machine because of the physical layout of the cluster.

5 IMPLEMENTION

In this chapter we will combine chapters 3 and 4 to show how we implemented our solution on a user mapping and permission service using ARC, see Figure 5.

5.1 Argus

We used the installation and configuration manuals, that can be found on the (Argus Authorization Service, 2013), and YAIM (YAIM Ain't an Installation Manager) that aims to simplify the installation and configuration process of grid services (What is YAIM, 2009). We found that in the end you still need to read the full documentation to fullfil the installation in either case to be able to create a good configuration. PDP was installed and configured without problems. The PEP configuration file in APPENDIX 7 was useful to read when we were troubleshooting the setup and also to change the "adminPassword".

5.1.1 PAP

The configuration is done in the Policy Administration Point (PAP). This is most likely the part of Argus where an administrator will spend most of his time. In the file /etc/ar-

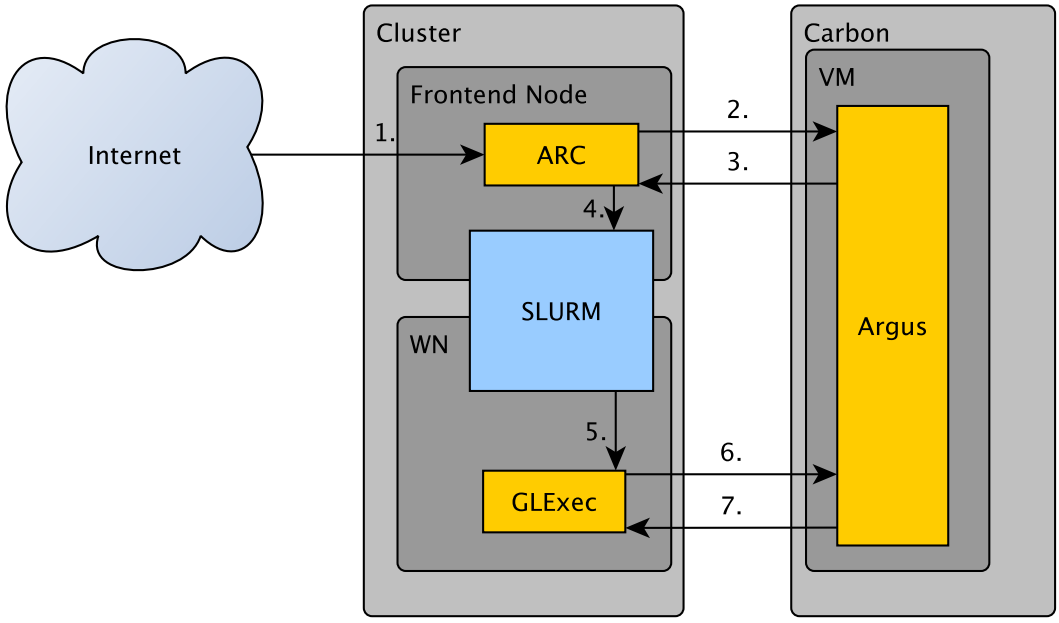

Figure 5. The software and hardware combined

gus/pap/pap_authorization.ini (APPENDIX 6) we added the DN (Distinguished Name) for the personal certificates that we wanted to have administration access to the PAP from the PAP-admin tool. Having root access to the system doesn't mean that you have admin access to PAP. The PAP configuration is shown in APPENDIX 5.

5.1.2 Implementing the Policies with PAP-admin CLI

One can use the PAP-admin CLI (Command Line Interface) to administrate PAP (Argus Policy Administration Point , PAP). We have created a file with the policies that we wanted to implement, see APPENDIX 4. When we had created the file we added the policies to PAP with the command "pap-admin add-policies-from-file <file name>".

5.1.3 The PAP Policies File

We will go through the rows in APPENDIX 4. The row "resource" tells which resource these rules are valid for. The star "*" means any resources, but it would be possible to create different rules for different clusters. The row

```
obligation "http://glite.org/xacml/obligation/local-environment-map"
```

is mandatory if one wants to implement user mapping (The Simplified Policy Language, 2010), as this will make Argus take care of the user mapping.

The row 'action ".*" ' stands for what action the user applies permission for. One could implement different actions, for example; access to storage, access to cluster etc. We use a star "*" to symbolise any action because we only use Argus to make authorisation decisions if users have permission to run jobs at our site.

The rows "rule permit" tell us which parameters we want Argus to permit. We have specified that we want to permit the VOs "cms", "ops" and "dteam". After the "rule

permit" rows we have the row 'rule deny vo="*" ' which specifies that we deny all other VOs. Because Argus reads the rules from top to down, the "rule deny" is last and therefore the rows "rule permit" will get executed first. Once an access decision is made Argus will stop trying to find a new access policy. One could also make rules based on FQAN or certificate subjects. If one would send a proxy certificate without a VO Argus would not be able to make a "Permit" or "Deny" decision and would return "Indeterminate".

5.1.4 Keeping Argus Functioning

It is important that the clocks are correct because proxy certificates have a short time window of validity. We have realized that virtual machines have a tendency to get their clocks out of sync. To keep the clocks up to date we use NTP (Network Time Protocol). To keep accepted certificates and VOMS up to date we have configured YUM (Yellowdog Updater, Modified) to install updates automatically.

5.2 Local Users and Configuration

To be able to use Argus as a user mapping service the same user names must be configured in Argus, in SLURM, and on the clusters. On a cluster all usernames are added by the bash script in APPENDIX 12 on the fronted node. The script adds users and gives them appropriate home directories. The home directories are shared in the cluster over NFS. We are using a consistent UNIX uid so that the that same configuration might work if our Tier sites storage upgrade to NFS4 even if it there is no plan for that at the moment. The UNIX users are synchronised by NIS from the fronted node to all the WNs on the clusters.

5.2.1 Configuring User Mapping for Argus

To be able to add user mapping capability to Argus one must first create a list of possible users. I wrote a script to create the list in Argus (APPENDIX 8). Because there will be less UNIX users than possible grid users Argus will reuse UNIX user names when Argus has mapped grid users to all available local users. Note that Argus uses UNIX names and not UNIX uids.

There are different types of groups and users that SLURM will give different priorities to. In the "voms-grid-mapfile" the decision is made on which VOMS role will be mapped to which user, see APPENDIX 9. For example a proxy certificate that has the VOMS role "/cms/Role=production" can be mapped to any user named userprod<R>, where <R> is a number between 001 and 200 see APPENDIX 8 and APPENDIX 9. There is also a "groupmapfile" that has the same function as the "voms-grid-mapfile" but instead of UNIX user basis it is on user group basis. The user group has no practical purposes in this thesis work, because on the clusters the users will be mapped to users in the primary UNIX group name.

5.2.2 Adding Users to SLURM

The job must be owned by a user that is configured in SLURM. The user must belong to a group. The group must be a part of a Partition. A Partition is called a queue. A queue must have some worker nodes allocated to it, for jobs to run on. The three rows below in slurm.conf on Alcyone-install are important for our setup:

```
PartitionName=gridcms Nodes=al25,alg[01-20] Default=NO
MaxNodes=4 DefaultTime=3-0 MaxTime=4-0 Shared=NO Priority=1
AllowGroups=cmsops,cmsuser,cmsprio,cmsprod,cmssgm State=Up
```

PartitionName stands for our queue "gridcms". Nodes stand for all worker nodes that are assigned to our queue, a total of 21 worker nodes. AllowGroups are all SLURM groups that are allowed to be in this queue. I have configured the SLURM groups to have the same name as the UNIX groups. To add users to these groups I created the script in APPENDIX 15. It would probably have been more reliable to create a configuration file and add the configuration file with the "sacctmgr load file=<file-name>" command (SACCTMGR, 2009) instead of adding users by a script.

5.3 ARC Communication with Argus

As in the previous chapter is stated, ARC is using the wrapper script in APPENDIX 1 to communicate with Argus. ARC is calling the wrapper script twice as can be seen in APPENDIX 2 under the headlines "[grid-manager]" and "[gridftp]".

5.3.1 Authorisation

The authorisation row can be seen in APPENDIX 2 as 'authplugin="ACCEPTED timeout=20 /root/scripts/pepcli_wrapper.py %C/job.%I.proxy" '

authplugin tells what authorisation method we should use. "ACCEPTED" is the ARC state in which the plug-in runs. The timeout=20 tells how many seconds ARC will wait before it fails. The file "/root/scripts/pepcli_wrapper.py" is the wrapper script and '%C/job.%I.proxy"' defines the proxy certificates location, which we want to check if it is authorised (F. Paganelli, 2015).

5.3.2 User Mapping

The user mapping row in APPENDIX 2 is ’unixmap="* mapplugin 20 /root/scripts/pepcli_wrapper.py%P ’

"Unixmap " is a way of defining the UNIX user of a grid job. "mapplugin" tells ARC to run an external script, "20" is the timeout in seconds, "/root/scripts/pepcli_wrapper.py" is the wrapper script. %P is the name of the proxy certificate (F. Paganelli, 2015).

5.4 GLExec

GLExec is installed on every WN see APPENDIX 10 and 11 for the configuration. GLEexc is installed by the kickstart file (Kickstarting a Machine, 2015) that our clusters are using. See APPENDIX 12 to find out what is done during the installation process. It is important that GLExec’s files have the right permissions and ownerships, see APPENDIX 12. In APPENDIX 10 it is important to note that all users, that pilot jobs get mapped to, are in the white_list because pilot jobs will execute GLExec and they can only execute GLExec if they are white listed.

6 TESTING

There are a lot of sites that are monitoring the status of different Tier-sites. The first warnings come from these monitoring sites if something is not working.When one is trying to find the source of a problem it is a good idea to follow the information flow as well as the job, and to check the logs where the first sign of the problem occurred. To test if a cluster is working the command arctest (ARC client installation instructions, 2015) can be used.

6.1 ARC

The first thing to test is if the ARC services are running:

```
service gridftpd status
service a-rex status
service nordugrid-arc-ldap-infosys status
service nordugrid-arc-inforeg status
service acix-cache status
```

The gridftpd.log is useful for troubleshooting the user mapping configuration. The log can be found on the grid node, default in /var/log/arc/gridftpd.log. A extended sample of the log is shown in APPENDIX 14.

Below is an example of a working setup. The two first lines show that the connection is established to ARC, the third line shows the proxy certificate's subject line.

```
[Arc.GridFTP_Commands] [INFO] [8541/38822480] Accepted connection on 128.214.177.171:2811
[Arc.GridFTP_Commands] [INFO] [8541/38822480] Accepted connection from 128.142.189.40:31113
[Arc.GridFTP_Commands] [INFO] [8541/140338660882336] User subject:
/DC=ch/DC=cern/OU=computers/CN=cmspilot12/vocms0167.cern.ch
[Arc.GridFTP_Commands] [INFO] [8541/140338660882336] Encrypted: true
```

The first thing ARC does is mapping the grid job to a local user named "gridnull". Thereafter ARC calls ARGUS through the PEPCLI wrapper script and Re-maps the user to the user name ARGUS replays with.

```
[Arc.userspec_t] [INFO] [8541/140338660882336] Initially mapped to local user: gridnull
[Arc.userspec_t] [INFO] [8541/140338660882336] Mapped to local id: 514
[Arc.userspec_t] [INFO] [8541/140338660882336] Mapped to local group id: 527
[Arc.userspec_t] [INFO] [8541/140338660882336] Mapped to local group name: gridnull
[Arc.userspec_t] [INFO] [8541/140338660882336] Remapped to local user: cmsprod028
[Arc.userspec_t] [INFO] [8541/140338660882336] Remapped to local id: 1229
[Arc.userspec_t] [INFO] [8541/140338660882336] Remapped to local group id: 766
[Arc.userspec_t] [INFO] [8541/140338660882336] Remapped user's home: /home/cmsgrid/cmsprod
```

If the re-mapping fails there is a good chance that the reason can be found in the wrapper script's log. The location of the log can be seen in APPENDIX 1.

```
[Arc.FileRoot] [INFO] [8541/140338660882336] Registering directory: jobs with plugin:
jobplugin.so
[Arc.DirectFilePlugin] [INFO] [8541/140338660882336] Mount point home/cmsgrid/sessiondir
[Arc.JobPlugin] [INFO] [8541/140338593373024] Cleaning job
hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm
[Arc.GridFTP_Commands] [INFO] [8541/140338593373024] Closing connection
[Arc.GridFTP_Commands] [INFO] [8541/140338660882336] Closed connection
[Arc.gridftpd] [INFO] [8541/38822480] Child exited
```

The long random string is the job's unique name. One can find different files that belong to the job in /var/spool/nordugrid/jobstatus/. The files that belong to the job should be owned by the local user (in this case cmsprod028) of the job and the file names should begin with "job.UNIQUE-NAME". Especially the job.UNIQUE-NAME.failed file can contain useful information.

```
$ ls -l /var/spool/nordugrid/jobstatus/ |grep  hmYNDmFX5zlnRDG |awk '{print $3, $4, $9}'
cmsprod028 cmsprod job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.description
cmsprod028 cmsprod job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.diag
cmsprod028 cmsprod job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.errors
cmsprod028 cmsprod job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.failed
cmsprod028 cmsprod job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.grami
cmsprod028 cmsprod job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.input
cmsprod028 cmsprod job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.local
cmsprod028 cmsprod job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.output
cmsprod028 cmsprod job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.proxy
root root job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.statistics
root root job.hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm.xml
```

The /var/log/arc/gm-jobs.log is also useful especially when you want to check if ARC is submitting the jobs correctly to a SLURM queue. Below is a snippet from the gm-jobs.log. You can see that the UID and GID are the same as in gridftpd.log. You can also see which queue the jobs are submitted to, in this case it is "gridcms". Also the proxy certificate's owner identity is visible.

```
2015-04-08 16:49:22 Started - job id:
```

```
hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm,
unix user: 1229:766, name: "", owner:
"/DC=ch/DC=cern/OU=computers/CN=cmspilot12/vocms0167.cern.ch",
lrms: SLURM, queue: gridcms
```

It is also possible to make ARC write in more detail to the log files by changing the debug variables in arc.conf, see APPENDIX 2.

6.2 ARGUS

The first thing to check is if the PEPCLI wrapper script is outputting anything to the log.

If everything is working properly then the log should not receive any messages. There is two exceptions.

1. The wrapper script does not have execution permission.

2. The wrapper script does not have permission to write to the log file.

These are the most common errors that the wrapper script writes to the log:

- "timeleft : 0:00:00", means that the proxy certificate has expired.

- "Decision: Deny" means that somebody that does not have access has tried to submit a job to the cluster and ARGUS has denied access.

- "failed: curl[7] Couldn't connect to server" means that there is something wrong with the connection, the URL is wrong, a firewall is restricting connection, or something similar.

- "Decision: Not Applicable" means that Argus does not have a rule that will either Permit or Deny job access. This is inconsistent from an administration point of view, there should be a rule in place to either Deny or Accept all requests. I suggest review of ARGUS rules with the pap-admin tool .

- "Indeterminate" shows up frequently in the log, then the clock is probably incorrect on the Argus server.

On the ARGUS server ARGUS is writing heavily to the log files in /var/log/argus/ but I have not found these log files helpful. In troubleshooting they seem to be more useful for record keeping, one can for example in /var/log/argus/pepd/process.log easily see who has been mapped to which unix user name.

6.3 Glexec

GLexec has good man pages to help with trouble shooting (Man Glexec 2010; Man Glexec.conf 2010)

There are two error codes that I came across when I was making the configuration; 202 and 203. 202 is most likely caused by wrong permission bits for the executable and the configuration file. The GLExec executable is supposed to have the permission 4711 and glexec.conf must have the permission 0400.

```
$ ls -l /usr/sbin/glexec; ls -l /etc/glexec.conf
-rws--x--x. 1 root root 144272 May 16  2012 /usr/sbin/glexec
-r--------. 1 root glexec 904 Jan  5 10:02 /etc/glexec.conf
```

The error 203 tells that there is most likely a configuration error in glexec.conf or that the user has not permission to use GLExec. The user that tried to execute GLExec is then not white listed in glexec.conf.

6.4 SLURM

SLURM has on the Install node a good log in /var/log/slurm/slurm_jobcomp.log, where you can find information about where jobs are running.

```
JobId=2397189 UserId=cmsprod003(1204) GroupId=cmsprod(766) Name=gridjob
JobState=COMPLETED Partition=gridcms TimeLimit=30 StartTime=2015-03-23T04:34:00
EndTime=2015-03-23T04:34:44 NodeList=alg15 NodeCnt=1 ProcCnt=1 WorkDir=
/home/cmsgrid/sessiondir/krWNDmBM7tlnRDGs8pg5lGMpABFKDmABFKDmVTKKDmABFKDmUEyKHm
```

Some useful commands to troubleshoot SLURM are sinfo, sacct, squeue.

It is important that all jobs are submitted to a queue. If a job is not submitted to a queue it means that the job will never run. This can be hard to troubleshoot because I have not found a good error message that warns for this problem. The easiest way to detect this problem is by using the squeue command to show all jobs and their queues. It is equally important that all queues have some working nodes attached to them. If a queue has no working nodes, it means that SLURM will submit jobs to the queue but can not get the jobs executed.

7 CONCLUSIONS

I have created a working environment, which was a goal of this thesis work. Once one has learned to master the environment it quite easy to make changes and create more advanced configurations. The biggest challenge is the steep learning curve. The administrator must have knowledge of all components, of certification, and of the features of grid jobs in order to be able to manage the envirorment.

7.1 Possible Improvements

ARC's configuration should be revised. Specifications which are no longer valid should be removed. This is probably true for the configuration of all components.

ARGUS could possibly be used by all grid nodes at UH. Then access policies at the clusters would be more easily managed. One problem with ARGUS is that it provides a single point of failure. It would therefore be a good idea to have a second installed and configured ARGUS server, even if it is fairly easy to create a new ARGUS server from scratch.

REFERENCES

About. 2014. Available: https://www.scientificlinux.org/about/, Retrieved: 14.01.2015.

About Advanced Resource Connector. 2015. Available: http://www.nordugrid.org/arc/about-arc.html, Retrieved: 02.02.2015.

About CERN. 2013. Available: http://home.web.cern.ch/about, Retrieved: 10.07.2014.

Alcyone Cluster. 2013. Available: http://docs.physics.helsinki.fi/alcyone.html, Retrieved: 04.03.2015.

Annual Report 2013 Helsinki Institute of Physics. 2014. Available: http://www.hip.fi/wp-content/uploads/2013/09/HIP-Annual-Report-2013.pdf, Retrieved: 26.04.2015.

ARC client installation instructions. 2015. Available: http://www.nordugrid.org/documents/arc-client-install.html, Retrieved: 20.04.2015.

arcproxy(1) - Linux man page(1). 2015. Available: http://linux.die.net/man/1/arcproxy, Retrieved: 19.02.2015.

Argus Authorization Service. 2013. Available: https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework, Retrieved: 16.02.2015.

Argus Policy Administration Point (PAP): Administration. 2012. Available: https://twiki.cern.ch/twiki/bin/view/EGEE/AuthZPAPCLI, Retrieved: 13.12.2014.

Argus System Administrator Guide. 2013. Available: http://argus-authz.github.io/doc/EMI-Argus-SysAdminGuide-1.1.0.pdf, Retrieved: 03.03.2015.

Background Reference: What are Certificates and PKI? 2015. Available: https://docs.puppetlabs.com/background/ssl/certificates_pki.html, Retrieved: 24.02.2015.

CMS. 2014. Available: http://home.web.cern.ch/about/experiments/cms, Retrieved: 10.07.2014.

The CMS Computing Project Technical Design Report. 2005. Available: http://cmsdoc.cern.ch/cms/CMG/CTDR/lhcc-2005-023.pdf, Retrieved: 10.7.2014.

Computing Grid. 2011. Available: http://cms.web.cern.ch/news/computing-grid, Retrieved: 10.07.2014.

F. Paganelli, O. Smirnova, Zs. Nagy. 2015, *ARC Computing Element System Administrator Guide*.

gLExec Deployment Tracking. 2015. Available: https://twiki.cern.ch/twiki/bin/view/LCG/GlexecDeploymentTracking, Retrieved: 21.04.2015.

gLExec-wn. 2015. Available: http://www.eu-emi.eu/products/-/asset_publisher/1gkD/content/glexec-wn-1, Retrieved: 08.01.2015.

The Grid: Software, middleware, hardware. 2015. Available: http://home.web.cern.ch/about/computing/grid-software-middleware-hardware, Retrieved: 20.01.2015.

Homepage of the Linux NIS/NIS+ Projects. 2012. Available: http://www.linux-nis.org, Retrieved: 20.01.2015.

How to configure VOMS LSC files. 2014. Available: https://twiki.cern.ch/twiki/bin/view/LCG/VOMSLSCfileConfiguration, Retrieved: 19.02.2015.

Introduction. 2015. Available: http://grid-deployment.web.cern.ch/grid-deployment/glite-web/introduction, Retrieved: 20.01.2015.

Kernel Based Virtual Machine. 2015. Available: http://www.linux-kvm.org/page/Main_Page, Retrieved: 15.02.2015.

Kickstarting a Machine. 2015. Available: https://access.redhat.com/documentation/en-US/Red_Hat_Network_Satellite/5.3/html/Deployment_Gui Retrieved: 02.02.2015.

Koivumäk, Jesper. 2010, *Development of a Plugin for Transporting Jobs Through a Grid (2010)*, Degree thesis, Arcada – Nylands Svenska Yrkeshogskola.

The Large Hadron Collider. 2014. Available: http://home.web.cern.ch/topics/large-hadron-collider, Retrieved: 10.07.2014.

Man Glexec. 2010. Available: http://www.nikhef.nl/grid/lcaslcmaps/man/glexec.1.0_9.html, Retrieved: 14.04.2015.

Man Glexec.conf. 2010. Available: http://www.nikhef.nl/grid/lcaslcmaps/man/glexec.conf.5.0_9.html, Retrieved: 14.04.2015.

Manage virtual machines with virt-manager. 2013. Available: http://virt-manager.org, Retrieved: 02.02.2015.

munge. 2015. Available: https://code.google.com/p/munge/.

The NorduGrid Collaboration. 2015. Available: http://www.nordugrid.org/about.html, Retrieved: 20.01.2015.

What are Certificates? 2014. Available: https://gridca.cern.ch/gridca/Help/Contents/Files/CNL_CERNCA.pdf, Retrieved: 26.04.2015.

Overview. 2013. Available: http://slurm.schedmd.com/overview.html, Retrieved: 20.12.2014.

The NorduGrid architecture and tools. 2003. Available: http://www.nordugrid.org/documents/MOAT003.pdf, Retrieved: 04.05.2015.

Proxy Certificates. 2008. Available: http://egee-uig.web.cern.ch/egee-uig/beta_pages/ProxyCerts.html, Retrieved: 20.02.2015.

SACCTMGR. 2009. Available: https://computing.llnl.gov/linux/slurm/sacctmgr.html, Retrieved: 26.04.2015.

Setting Up KPI : Certificates and VOMS. 2015. Available: http://www.eu-emi.eu/training/cert-tutorial, Retrieved: 20.02.2015.

The Simplified Policy Language. 2010. Available: https://twiki.cern.ch/twiki/bin/view/EGEE/SimplifiedPolicyLanguage, Retrieved: 20.02.2015.

USG Proxy Certificates. 2013. Available: https://wiki.egi.eu/wiki/USG_Proxy_Certificates, Retrieved: 19.02.2015.

Virtual Organisations. 2015. Available: http://wlcg.web.cern.ch/getting-started/VO, Retrieved: 27.04.2015.

voms-proxy-init(1) - Linux man page. 2015. Available: http://linux.die.net/man/1/voms-proxy-init, Retrieved: 19.02.2015.

What are Certificates? 2013. Available: https://gridca.cern.ch/gridca/Help/?kbid=011010, Retrieved: 26.04.2015.

What is YAIM. 2009. Available: https://twiki.cern.ch/twiki/bin/view/EGEE/YAIM, Retrieved: 16.02.2015.

Why is CMS so big? 2011. Available: http://cms.web.cern.ch/news/why-cms-so-big, Retrieved: 10.07.2014.

Why make Scientific Linux? 2014. Available: https://www.scientificlinux.org/about/why-make-scientific-linux/, Retrieved: 15.02.2015.

APPENDICES

APPENDIX 1. PEPCLI WRAPPER SCRIPT

```
#!/usr/bin/python
import commands
import sys
import os

input = commands.getoutput( "/usr/bin/pepcli \
            --pepd https://argus-vm.hip.helsinki.fi:8154/authz \
            --certchain " + str(sys.argv[1]) + " \
            --cert /etc/grid-security/hostcert.pem \
            --key /etc/grid-security/hostkey.pem \
            --cacert /etc/grid-security/certificates/NorduGrid.pem \
            --resource 'SomeThing' --action 'someThingElse' " )

a = input.split('\n')
Name=""
for line in a:
    b = line.split('')
    if b[0] == "Username:":
       Name = b[1]
       elif b[0] == "Group:":
            word = Name + ":*"
            print word
            sys.exit(0)

wrt = open("/var/log/oscars-messages", "a")
wrt.write( str( commands.getoutput( "date" ) + " - " ) + \
      sys.argv[0] + str( " - " ) + input + "\n" )
wrt.write( str( commands.getoutput( "/usr/bin/voms-proxy-info \
      --all --file " + str( sys.argv[1] )) + str( "\n" )))
wrt.close()
print input
sys.exit(1)
```

APPENDIX 2. ARC.CONF ON ALCYONE WITHOUT COMMENTS

```
[common]
hostname="alcyone-cms.grid.helsinki.fi"
lrms="SLURM gridcms"
LRMSName="SLURM"
SLURM_bin_path="/usr/bin"
SLURM_wakeupperiod="15"
SLURM_wakeupperiod="60"
globus_tcp_port_range="9000,12000"
globus_udp_port_range="9000,12000"
x509_user_key="/etc/grid-security/hostkey.pem"
x509_user_cert="/etc/grid-security/hostcert.pem"
x509_cert_dir="/etc/grid-security/certificates"
gridmap="/etc/grid-security/grid-mapfile"
voms_processing="standard"
voms_trust_chains = "/O=Grid/O=NorduGrid/CN=host/voms.fgi.csc.fi /O=Grid/O=NorduGrid/
CN=NorduGrid Certification Authority ----NEXT CHAIN--- /DC=ch/DC=cern/OU=computers/
CN=lcg-voms.cern.ch /DC=ch/DC=cern/CN=CERN Trusted Certification Authority ----NEXT
CHAIN--- /DC=ch/DC=cern/OU=computers/CN=voms.cern.ch /DC=ch/DC=cern/CN=CERN Trusted
Certification Authority"
[vo]
id="ndgfops"
vo="ndgfops"
file="/etc/grid-security/grid-mapfile"
source="vomss://voms.ndgf.org:8443/voms/ops.ndgf.org"
mapped_unixid="gridnull"
[vo]
id="egiops"
vo="ops"
file="/etc/grid-security/grid-mapfile"
source="vomss://voms.cern.ch:8443/voms/ops"
```

```
[vo]
id="cms"
vo="cms"
file="/etc/grid-security/grid-mapfile"
source="vomss://voms.cern.ch:8443/voms/cms"
mapped_unixid="gridnull"
[ group]
[group/ndgfops]

name="ndgfops"
vo="ndgfops"
file="/etc/grid-security/grid-mapfile"
[group/cmsprod]
name="cmsprod"
voms="cms * production *"
file="/etc/grid-security/grid-mapfile"
[group/cmsprio]
name="cmsprio"
voms="cms * priorityuser *"
file="/etc/grid-security/grid-mapfile"
[group/cmsops]
name="cmsops"
voms="ops * * *"
file="/etc/grid-security/grid-mapfile"
[group/cmsuser]
name="cmsuser"
vo="cms"
file="/etc/grid-security/grid-mapfile"
[grid-manager]
authplugin="ACCEPTED timeout=20 /root/scripts/pepcli_wrapper.py \%C/job.\%I.proxy"
controldir="/var/spool/nordugrid/jobstatus"
sessiondir="/home/cmsgrid/sessiondir"
runtimedir="/home/cmsgrid/runtime"
cachedir="/home/cmsgrid/cache"
```

```
cachesize="80 70"
cachelifetime="30d"
user="root"
debug="3"
logsize="1000000 52"
shared_filesystem="yes"
mail="alcyone-admin@helsinki.fi"
joblog="/var/log/arc/gm-jobs.log"
maxjobs="10000 1000 400"
maxload="20 2 10"
wakeupperiod="30"
defaultttl="1209600"
authplugin="FINISHED timeout=30,onfailure=pass /usr/libexec/arc/arc-ur-logger %C %I %S %U"
maxtransfertries="10"
arex_mount_point="https://alcyone-cms.grid.helsinki.fi:443/arex"
[gridftpd]
unixmap="* mapplugin 20 /root/scripts/pepcli_wrapper.py\%P"
debug="3"
logsize="1000000 52"
user="root"
encryption="yes"
x509_user_key="/etc/grid-security/hostkey.pem"
x509_user_cert="/etc/grid-security/hostcert.pem"
x509_cert_dir="/etc/grid-security/certificates"
gridmap="/etc/grid-security/grid-mapfile"
allowunknown="yes"
unixgroup="ops simplepool /etc/grid-security/cmsopspool/"
unixgroup="cmsuser simplepool /etc/grid-security/cmsuserpool/"
unixgroup="cmsprod simplepool /etc/grid-security/cmsprodpool/"
unixgroup="ndgfops simplepool /etc/grid-security/cmsopspool"
[gridftpd/jobs]
path="/jobs"
plugin="jobplugin.so"
allownew="yes"
```

```
maxjobdesc="5242880"
[infosys]
port="2135"
debug="1"
timelimit="1800"
registrationlog="/var/log/arc/inforegistration.log"
providerlog="/var/log/arc/infoprovider.log"
provider_loglevel="2"
infosys_nordugrid="enable"
infosys_glue12="enable"
infosys_glue2_ldap="disable"
[infosys/admindomain]
name="grid.helsinki.fi"
[infosys/glue12]
resource_location="Helsinki, Finland"
resource_latitude="60.1915"
resource_longitude="24.8993"
cpu_scaling_reference_si00="4051"
processor_other_description="Cores=12,Benchmark=15.8-HEP-SPEC06"
glue_site_web="http://www.helsinki.fi/yliopisto/"
glue_site_unique_id="FI_HIP_T2"
provide_glue_site_info="false"
[infosys/cluster/registration/fgigiis2]
targethostname=giis2.fgi.csc.fi
targetport="2135"
targetsuffix="mds-vo-name=Finland,o=grid"
regperiod="300"
registranthostname="alcyone-cms.grid.helsinki.fi"
registrantport="2135"
[infosys/cluster/registration/fgigiis1]
targethostname=giis1.fgi.csc.fi
targetport="2135"
targetsuffix="mds-vo-name=Finland,o=grid"
registranthostname="alcyone-cms.grid.helsinki.fi"
```

```
regperiod="300"
registrantport="2135"
[cluster]
cluster_alias="Alcyone (CMS)"
comment="FGI cluster (CMS)"
cluster_location="FI-00014"
cluster_owner="University of Helsinki"
authorizedvo="cms"
authorizedvo="ops"
localse="madhatter.csc.fi"
nodeaccess="outbound"
clustersupport="alcyone-admin@helsinki.fi"
homogeneity="False"
architecture="adotf"
defaultmemory="1960"
opsys="CentOS"
opsys="6.3"
benchmark="SPECINT2000  4051"
benchmark="HEPSPEC2006 15.8"
[queue/gridcms]
name="gridcms"
homogeneity="False"
comment="CMS grid queue"
nodecpu="Intel(R) Xeon(R) CPU X5650  @ 2.67GHz"
nodememory="1960"
defaultmemory="1960"
benchmark="SPECINT2000  4051"
benchmark="HEPSPEC2006 15.8"
```

APPENDIX 3. GLEXEC.CONF

```
#
# Glexec configuration file
#
[glexec ]
silent_logging = no
log_level = 3
user_white_list = glexec,.cmsuser,.cmsp,.cmspriority,.ndgfops,.cmsprod,cmssgm
group_white_list = cmsuser,glexec
linger = yes
user_identity_switch_by = glexec
use_lcas = no
lcmaps_db_file = /etc/lcmaps/lcmaps-glexec.db
lcmaps_log_file = /var/log/glexec/lcas_lcmaps.log
lcmaps_debug_level = 0
lcmaps_log_level = 1
lcmaps_get_account_policy = glexec_get_account
lcmaps_verify_account_policy = glexec_verify_account

lcas_db_file = /etc/lcas/lcas-glexec.db
lcas_log_file = /var/log/glexec/lcas_lcmaps.log
lcas_debug_level = 0
lcas_log_level = 1
preserve_env_variables = no
log_destination = syslog
```

APPENDIX 4. ARGUS POLICIES

```
default (local):

resource ".*" {
    obligation "http://glite.org/xacml/obligation/local-environment-map" {
    }

    action ".*" {
        rule permit { vo="cms" }
        rule permit { vo="ops" }
        rule permit { vo="dteam" }
        rule deny { vo="*" }
    }
}
```

APPENDIX 5. PAP_CONFIGURATION.INI

```
#
# Configuration file created by YAIM on 04/12/2012 15:43
#
# PAP configuration
#
# Documentation: https://twiki.cern.ch/twiki/bin/view/EGEE/AuthZPAPConfig
#

[paps]
# Trusted PAPs will be listed here

[paps:properties]
poll_interval = 14400
ordering = default

[repository]
location = /usr/share/argus/pap/repository
consistency_check = false
consistency_check.repair = false

[standalone-service]
entity_id = http://argus-vm.hip.helsinki.fi/pap
hostname = argus-vm.hip.helsinki.fi
port = 8150
shutdown_port = 8151
shutdown_command = papshutdown_20121204154256

[security]
certificate = /etc/grid-security/hostcert.pem
private_key = /etc/grid-security/hostkey.pem
```

APPENDIX 6. PAP_AUTHORIZATION.INI

```
#
# Configuration file created by YAIM on 04/12/2012 15:43
#
# PAP service access control
#
# Documentation: https://twiki.cern.ch/twiki/bin/view/EGEE/AuthZPAPConfig
#
#Oscar added:
[dn]
"/DC=org/DC=terena/DC=tcs/OU=Domain Control Validated/CN=argus-vm.hip.helsinki.fi" : ALL
"/DC=org/DC=terena/DC=tcs/C=FI/O=University of Helsinki/CN=Carl Kraemer" : ALL
"/O=Grid/O=NorduGrid/CN=host/argus-vm.hip.helsinki.fi" : ALL
[fqan]
```

APPENDIX 7. PEPD.INI

```
#
# Configuration file created by YAIM on 04/12/2012 15:43
#
# Argus PEP Server configuration
#
# Documentation: https://twiki.cern.ch/twiki/bin/view/EGEE/AuthZPEPDConfig
#
[SERVICE]
entityId = http://argus-vm.hip.helsinki.fi/pepd
hostname = argus-vm.hip.helsinki.fi
port = 8154
adminPort = 8155
adminPassword = ************************

pips = REQVALIDATOR_PIP OPENSSLSUBJECT_PIP GLITEXACMLPROFILE_PIP COMMONXACMLPROFILE_PIP
obligationHandlers = ACCOUNTMAPPER_OH

[PDP]
pdps = https://argus-vm.hip.helsinki.fi:8152/authz

[SECURITY]
servicePrivateKey = /etc/grid-security/hostkey.pem
serviceCertificate = /etc/grid-security/hostcert.pem
trustInfoDir = /etc/grid-security/certificates/
enableSSL = true
#Oscar has made these changes:
#requireClientCertAuthentication = false
requireClientCertAuthentication = true

#
# Policy Information Points (PIP) configuration
#
```

```
[REQVALIDATOR_PIP]
parserClass = org.glite.authz.pep.pip.provider.RequestValidatorPIPIniConfigurationParser
validateRequestSubjects = true
validateRequestResources = true
validateRequestAction = true
validateRequestEnvironment = false

[OPENSSLSUBJECT_PIP]
parserClass = org.glite.authz.pep.pip.provider.OpenSSLSubjectPIPIniConfigurationParser
opensslSubjectAttributeIDs = http://glite.org/xacml/attribute/subject-issuer
          urn:oasis:names:tc:xacml:1.0:subject:subject-id
opensslSubjectAttributeDatatypes = http://ww[w.w3.org/2001/XMLSchema#string

[GLITEXACMLPROFILE_PIP]
parserClass = org.glite.authz.pep.pip.provider.GLiteAuthorizationProfilePIPIniConfigurationParser
vomsInfoDir = /etc/grid-security/vomsdir/
acceptedProfileIDs = http://glite.org/xacml/profile/grid-ce/1.0 http://glite.org/xacml/profile/grid-wn/1.0

[COMMONXACMLPROFILE_PIP]
parserClass = org.glite.authz.pep.pip.provider.CommonXACMLAuthorizationProfilePIPIniConfigurationParser
vomsInfoDir = /etc/grid-security/vomsdir/
acceptedProfileIDs = http://dci-sec.org/xacml/profile/common-authz/1.1

#
# Obligation Handlers (OH) configuration
#
[ACCOUNTMAPPER_OH]
parserClass = org.glite.authz.pep.obligation.dfpmap.DFPMObligationHandlerConfigurationParser
handledObligationId = http://glite.org/xacml/obligation/local-environment-map
accountMapFile = /etc/grid-security/grid-mapfile
groupMapFile = /etc/grid-security/groupmapfile
gridMapDir = /etc/grid-security/gridmapdir
useSecondaryGroupNamesForMapping = true
```

APPENDIX 8. ADDUSERSTOARGUS.SH

```
#!/bin/bash
# Creates usermapping configuration for Argus 2014
#
# Oscar Kraemer 2014
#
for i in {1..9}
do
        touch /etc/grid-security/gridmapdir/cmsprio00$i
        touch /etc/grid-security/gridmapdir/cmsprod00$i
        touch /etc/grid-security/gridmapdir/cmsuser00$i
        touch /etc/grid-security/gridmapdir/cmsops0$i
done
for i in {10..40}
do
        touch /etc/grid-security/gridmapdir/cmsops$i
done
for i in {10..99}
do
        touch /etc/grid-security/gridmapdir/cmsprio0$i
        touch /etc/grid-security/gridmapdir/cmsprod0$i
        touch /etc/grid-security/gridmapdir/cmsuser0$i
done
for i in {100..200}
do
        touch /etc/grid-security/gridmapdir/cmsprio$i
        touch /etc/grid-security/gridmapdir/cmsprod$i
        touch /etc/grid-security/gridmapdir/cmsuser$i
done
```

APPENDIX 9. VOMS-GRID-MAPFILE

"/cms/Role=lcgadmin" cmssgm

"/cms/Role=lcgadmin/Capability=NULL" cmssgm

"/cms/Role=production" .cmsprod

"/cms/Role=pilot/Capability=NULL" .cmsprod

"/cms/Role=pilot" .cmsuser

"/cms/Role=hiproduction" .cmsprod

"/cms/Role=NULL/Capability=NULL" .cmsuser

"/cms" .cmsuser

"/ops" .cmsops

APPENDIX 10. GLEXEC CONFIGURATION FILE

```
#
# Glexec configuration file
#
[glexec]
silent_logging = no
log_level = 3
user_white_list = glexec,.cmsuser,.cmsp,.cmspriority,.ndgfops,.cmsprod,cmssgm
group_white_list = cmsuser,glexec
linger = yes
user_identity_switch_by = glexec
use_lcas = no
lcmaps_db_file = /etc/lcmaps/lcmaps-glexec.db
lcmaps_log_file = /var/log/glexec/lcas_lcmaps.log
lcmaps_debug_level = 0
lcmaps_log_level = 1
lcmaps_get_account_policy = glexec_get_account
lcmaps_verify_account_policy = glexec_verify_account

lcas_db_file = /etc/lcas/lcas-glexec.db
lcas_log_file = /var/log/glexec/lcas_lcmaps.log
lcas_debug_level = 0
lcas_log_level = 1
preserve_env_variables = no
log_destination = syslog
```

APPENDIX 11. GLEXEC-LCMAPS CONFIGURATION

```
# LCMAPS config file for glexec generated by YAIM: Mon Jun 10 14:54:17 EEST 2013
#

# where to look for modules
path = /usr/lib64/lcmaps

# module definitions
verify_proxy = "lcmaps_verify_proxy.mod"
                  " -certdir /etc/grid-security/certificates"
#                " --only-enforce-lifetime-checks"
#                " --allow-limited-proxy"

posix_enf = "lcmaps_posix_enf.mod"
            " -maxuid 1"
            " -maxpgid 1"
            " -maxsgid 32"

pepc        = "lcmaps_c_pep.mod"
              "--pep-daemon-endpoint-url https://argus-vm.hip.helsinki.fi:8154/authz"
              " -resourceid http://authz-interop.org/xacml/resource/resource-type/wn"
              " -actionid http://glite.org/xacml/action/execute"
              " -capath /etc/grid-security/certificates/"
              " -pep-certificate-mode implicit"
              " --use-pilot-proxy-as-cafile"
#             " --pep-c-debug"

glexec_get_account:
verify_proxy -> pepc
#pepc -> posix_enf
```

APPENDIX 12. GLEXEC INSTALLING SCRIPT

```
yum -y install emi-glexec_wn emi-wn yaim-glexec-wn

/**

**/

#Setup glexec.conf file //Oscar
if [ -f /mnt/conf/glexec.conf ]; then
     cp /mnt/conf/glexec.conf /etc/glexec.conf
     chown glexec:root /etc/glexce.conf
     chmod 0400 /etc/glexec.conf
     chown root:root /usr/sbin/glexec
     chmod 4711 /usr/sbin/glexec
fi

if [ -f /mnt/conf/lcmaps-glexec.db ]; then
     cp /mnt/conf/lcmaps-glexec.db /etc/lcmaps/lcmaps-glexec.db
fi
```

APPENDIX 13. USERADD.SH

```
#!/bin/bash
#Oscar Kraemer
#Add CMS Users tp Jade-install 2014

mkdir /home/cmsgrid
mkdir /home/cmsgrid/cmsuser
mkdir /home/cmsgrid/cmsops
mkdir /home/cmsgrid/cmsprio
mkdir /home/cmsgrid/cmsprod
mkdir /home/cmsgrid/cmssgm

chmod 755 /home/cmsgrid
chmod 755 /home/cmsgrid/cmsuser
chmod 755 /home/cmsgrid/cmsops
chmod 755 /home/cmsgrid/cmsprio
chmod 755 /home/cmsgrid/cmsprod
chmod 755 /home/cmsgrid/cmssgm

groupadd cmsopsAlcyone -g 763
groupadd cmsuserAlcyone -g 764
groupadd cmsprioAlcyone -g 765
groupadd cmsprodAlcyone -g 766

for i in {1..9}
do
  useradd cmsops0$i -d /home/cmsgrid/cmsops --gid 763
```

```
  useradd cmsuser00$i -d /home/cmsgrid/cmsuser --gid 27100 -G 764
  useradd cmsprio00$i -d /home/cmsgrid/cmsprio --gid 27101 -G 765
  useradd cmsprod00$i -d /home/cmsgrid/cmsprod --gid 27102 -G 766
done

for i in {10..40}
do
  useradd cmsops$i -d /home/cmsgrid/cmsuser --gid 763

done
for i in {10..99}
do
  useradd cmsuser0$i -d /home/cmsgrid/cmsuser --gid 27100 -G 764
  useradd cmsprio0$i -d /home/cmsgrid/cmsprio --gid 27101 -G 765
  useradd cmsprod0$i -d /home/cmsgrid/cmsprod --gid 27162 -G 766
done
for i in {100..200}
do
  useradd cmsuser$i -d /home/cmsgrid/cmsuser --gid 27100 -G 764
  useradd cmsprio$i -d /home/cmsgrid/cmsprio --gid 27101 -G 765
  useradd cmsprod$i -d /home/cmsgrid/cmsprod --gid 27162 -G 766
done
```

APPENDIX 14. GRIDFTPD.LOG

```
[2015-04-08 16:52:59] [Arc.GridFTP_Commands] [INFO] [8541/38822480]
Accepted connection on 128.214.177.171:2811
[2015-04-08 16:52:59] [Arc.GridFTP_Commands] [INFO] [8541/38822480]
Accepted connection from 128.142.189.40:31113
[2015-04-08 16:53:00] [Arc.GridFTP_Commands] [INFO] [8541/140338660882336]
User subject: /DC=ch/DC=cern/OU=computers/CN=cmspilot12/vocms0167.cern.ch
[2015-04-08 16:53:00] [Arc.GridFTP_Commands] [INFO] [8541/140338660882336] Encrypted: true
[2015-04-08 16:53:00] [Arc.userspec_t] [INFO] [8541/140338660882336]
Initially mapped to local user: gridnull
[2015-04-08 16:53:00] [Arc.userspec_t] [INFO] [8541/140338660882336]
Mapped to local id: 514
[2015-04-08 16:53:00] [Arc.userspec_t] [INFO] [8541/140338660882336]
Mapped to local group id: 527
[2015-04-08 16:53:00] [Arc.userspec_t] [INFO] [8541/140338660882336]
Mapped to local group name: gridnull
[2015-04-08 16:53:00] [Arc.userspec_t] [INFO] [8541/140338660882336]
Remapped to local user: cmsprod028
[2015-04-08 16:53:00] [Arc.userspec_t] [INFO] [8541/140338660882336]
Remapped to local id: 1229
[2015-04-08 16:53:00] [Arc.userspec_t] [INFO] [8541/140338660882336]
Remapped to local group id: 766
[2015-04-08 16:53:00] [Arc.userspec_t] [INFO] [8541/140338660882336]
Remapped user's home: /home/cmsgrid/cmsprod
[2015-04-08 16:53:00] [Arc.FileRoot] [INFO] [8541/140338660882336]
Registering directory: jobs with plugin: jobplugin.so
[2015-04-08 16:53:00] [Arc.DirectFilePlugin] [INFO] [8541/140338660882336]
Mount point home/cmsgrid/sessiondir
[2015-04-08 16:53:00] [Arc.JobPlugin] [INFO] [8541/140338593373024]
Cleaning job hmYNDmFX5zlnRDGs8pg5lGMpABFKDmABFKDmrwGKDmABFKDmSppAYm
[2015-04-08 16:53:50] [Arc.GridFTP_Commands] [INFO] [8541/140338593373024] Closing connection
[2015-04-08 16:53:50] [Arc.GridFTP_Commands] [INFO] [8541/140338660882336] Closed connection
[2015-04-08 16:53:50] [Arc.gridftpd] [INFO] [8541/38822480] Child exited
```

APPENDIX 15. ADD USERS TO SLURM

```
#!/bin/bash
# Add user to pool acccounts on Alcyone-cms 2014 by Oscar
#
# New pool will be/was added for cmsops
# Creates pool accounts for ARC.
#

for i in {1..9}
do
        sacctmgr create user name=cmsops0$i cluster=alcyone account=cmsops fairshare=1
        sacctmgr create user name=cmsprio00$i cluster=alcyone account=cmsprio fairshare=1
        sacctmgr create user name=cmsprod00$i cluster=alcyone account=cmsprod fairshare=1
        sacctmgr create user name=cmsuser00$i cluster=alcyone account=cmsuser fairshare=1
done

for i in {10..40}
do
        sacctmgr create user name=cmsops$i cluster=alcyone account=cmsops fairshare=1
done

for i in {10..99}
do
        sacctmgr create user name=cmsprio0$i cluster=alcyone account=cmsprio fairshare=1
        sacctmgr create user name=cmsprod0$i cluster=alcyone account=cmsprod fairshare=1
        sacctmgr create user name=cmsuser0$i cluster=alcyone account=cmsuser fairshare=1
done

for i in {100..200}
do
        sacctmgr create user name=cmsprio$i cluster=alcyone account=cmsprio fairshare=1
        sacctmgr create user name=cmsprod$i cluster=alcyone account=cmsprod fairshare=1
        sacctmgr create user name=cmsuser$i cluster=alcyone account=cmsuser fairshare=1
done
```

APPENDIX 16. SAMMANFATTNING

WLCG (Worldwide Large hydron collider Computing Grid) beslutade att GLExec skall införas på alla Tier-2 siter. GLExec används för att kunna registrera ägaren av grid beräknings job. Vid Forskningsinstitutet för fysik (HIP) använder vi oss av ARC (Advanced Resource Connector), Argus, Slurm (Simple Linux Utility for Resource Management) samt GLExec för att skapa en fungerande lösning för åtkomstkontroll och användarregistrering.

HIPs Tier-2 site består huvudsakligen av tre stycken beräknings kluster som tillsammans har runt 2000 processor kärnor.

Pilotberäkningsuppgifter

Pilotberakningsuppgifter är en typ av grid jobb som inte utför beräkningar. Pilotberäkningsuppgifterna används för att köa in till kluster, när de har kommit in på klustret så kontrollerar de om klustrets miljö är funktions duglig. Om miljön är funktions duglig så drar pilotberakningsuppgifter in ett beräknings jobb som sedan exekveras på klustret beräknings noder.

Certifiering

För at kunna skicka grid jobb till en WLCG beräknings resurs krävs det att man har ett PKI (Public key infrastructure) certifikat som är signerat av en lämplig Certifikatutfärdare (CA) och att man tillhör en av WLCGs Virtuella Organisationer (VO). När ett grid jobb sänds skickas det med ett proxy certifikat som används för att identifiera jobbets ägare samt i vissa fall tillåta jobbet få tillgång till tjänster som kräver certifiering när den väl har anlänt till beräknings resursen.

Proxy certifikaten är ett certifikat som baseras på ägarens personliga certifikat. Skillnaden mellan det personliga certifikatet är att proxy certifikatet har en kortare giltighetstid, normalt 12 timmar. Den korta giltighetstiden beror på att proxy certifikatet är osäkert än det personliga certifikatet för att det sänds över internet och kan lättare bli stulet än ett personligt certifikat, så om det blir stulet så har förövaren en begränsad tid att göra skada.

Mjukvarukoponenter

För att skapa en fungerande lösning så används vi oss av flera mjukvarukomponenter.

Arc

Arc är ett grid middleware, som fungerar som dörren mellan internet och klustret. Då ett gridjob skickas till ett kluster som använder sig av ARC, så först anländer grid jobbet till ARC som tar emot jobbet. Efter att ARC tar emot jobbet så sparar ARC information om jobbet samt försöker ge jobbet en local UNIX användare. För att kunna registrera jobbet till en UNIX användare använder sig ARC av Argus. Efter att ARC har gett en UNIX användare åt jobbet skickas jobbet till en Slurm kö där jobbet väntar tills det finns en ledig beräknings kapacitet för jobbet att exekveras.

Argus

Argus är ett centraliserat åtkomst och användarregistrerings tjänst. Vid HIP använder vi oss av Argus för att bestämma om ett grid jobb har tillgång till våra kluster eller inte. Vi använder även Argus för att bestämma vilket UNIX namn jobbet skall få på basen av till vilka attributer jobbets användare har. Attributerna är bestämda inom VO:n som användaren hör till.

GLExec

GLExec används på beräknings noderna för att byta användaren på jobben som dras in av pilotberäkningsuppgifterna. GLExec behövs för att vi skall kunna vara säkra på att jobben pilotberäkningsuppgifterna drar in har rätt att exekveras på våra kluster. GLExec använder sig även av samma Argus tjänst som ARC.

Slurm

Slurm är en jobb schemaläggare den tar emot jobb försöker hitta en lämplig beräknings nod för jobbet att exekveras på då ett tidigare jobb blir färdigt. Slurm är en beräknings resurs specifik tjänst och kan användas direkt av lokala användare. Beroende på vad för UNIX användare jobbet har så bestäms prioriten på jobbet.

Kommunikation mellan komponenterna

Arc tar emot grid jobbet då det kommer från Internet, sedan använder ARC sig av ett wrapper skript som finns i APPENDIX 1. För att kontrollera om grid jobbets användare har rätt att använda klustret. Om användaren har rätt att använda klustret, gör ARC en till förfrågan med wrapper skriptet till Argus om till vilken UNIX användaren jobbet skall registreras till. När jobbet har fått ett UNIX användare namn så skickas jobbet till Slurm där det köar tills grid jobbet får tillgång till en beräknings nod där den exekveras.

Om gridjobbet är ett pilotberäkningsuppgifter så testar pilotberäkningsuppgifter beräknings resursen om den tycks vara funktions duglig. Om noden är funktions duglig så drar pilotberäkningsuppgifter in ett grid jobb. Pilotberäkningsuppgiften kallar på GLExec med det nya grid jobbets proxy certifikat. GLExec tar certifikatet och skickar en förfrågan till Argus om till vilken användare jobbet skall registeras till, om jobbet har rätt att exekveras

så svarar Argus med ett användarnamn. GLExec byter UNIX användare på gridjobbet till den nya UNIX användaren och jobbet exekveras med det nya UNIX användaren.

Slutsatser

I arbetet beskrivs hur man kan implementera en lösning för åtkomstkontroll och användarregistrering med komponenterna ARC, Argus, Slurm och GLExec för en WLCG tier-2 site. Jag lyckades att göra implementationen för HIP inom utsatt tidsgräns för att testerna för GLExcec klassades som kritiska. Vid utförande det praktiska delarna av arbetet hade ingen skapat en lösning med samma komponenter som jag var medveten om.