Building a Demilitarized Zone with Data Encryption for Grid Environments

Matthias Schmidt, Matthew Smith, Niels Fallenbeck, Hans Picht, Bernd Freisleben

Department of Mathematics and Computer Science, University of Marburg
Hans-Meerwein-Strasse, D-35032 Marburg, Germany
{schmidtm, matthew, fallenbe, picht, freisleb}@informatik.uni-marburg.de

ABSTRACT

Security and data integrity are important aspects in the fields of Grid and cluster computing. When these two areas are combined, the security issues intermingle and new security concepts are needed to ensure protection of both Grid users and local cluster users. In this paper, a novel dual laned Demilitarized Zone (DMZ) to protect local clusters from Grid attacks is introduced. The Globus Security Infrastructure (GSI) is extended to enable safe end-to-end encryption of Grid jobs through the DMZ and into virtualized execution hosts. Finally, an integrated Network Intrusion Detection System with Grid-specific rules, further protecting the Grid DMZ, is presented.

Keywords

Grid computing, cluster computing, network-level security and protection, site security monitoring, cryptography

1. INTRODUCTION

The Grid computing paradigm is aimed at providing resources (such as compute clusters, data, access to special appliances and even people) as easy as electricity is provided through the electrical power Grid. This necessitates that the Grid must be easy and transparent to access and use. Unlike traditional cluster computing in which only a small number of users work in a closed system, Grid computing exposes local clusters to a large number of users via the Internet using open Grid middlewares such as Globus, gLite and Unicore. Like most complex IT systems, these middleware solutions exhibit a number security problems [7, 8, 9, 10] opening the entire system to attack. Unfortunately, these security holes do not only expose Grid users to attack, but also existing cluster users who up until now have worked in a local and secure environment. This changing nature of Grid and cluster computing and the new threats are discussed in [20, 21].

Consequently, Grids are an attractive target for intruders, since standardized access to a large number of machines, which can be misused in various ways, is offered. The computing power of clusters exposed via the Grid can be used to break passwords and the large storage capacity is perfect for storing and sharing illegal software and data. The generous bandwidth of the Internet connection is ideal for launching Denial-of-Service (DoS) attacks or for hosting file sharing services, to name just a few attacks. However, more critical than these resource attacks are the attacks against customer data. If a Grid resource provider cannot ensure end-to-end integrity and safety of customer data, an industrial adoption of Grid technology will not be possible.

This leads to two major requirements a Grid system must fulfill if a widespread integration of existing cluster systems and an industrial adoption of Grid technologies is desired. First, existing cluster environments need to be isolated from the weaknesses of the Grid middleware and possibly unknown and malicious Grid users, while at the same time offering their compute power to the Grid. Second, Grid user data must be protected from malicious users both during transport and during computation on the backend clusters.

In this paper, we present a Grid security solution which isolates the compute clusters from the Grid headnode through a novel Grid enabled Demilitarized Zone (DMZ) and extends the Globus Security Infrastructure (GSI) to incorporate end-to-end encryption through the DMZ into virtual compute nodes on the backend cluster. A DMZ Grid Job Manager is introduced to ensure a secure transport of job data between the Grid headnode and the Cluster headnode. Thus, all customer related data is fully encrypted during all network transports and during storage on insecure locations like the Grid headnode. The use of virtualization technology allows the execution of jobs inside an isolated environment. Furthermore, the security of the DMZ is extended through a Network Intrusion Detection System (NIDS) to detect several Grid-specific attacks, especially Denial-of-Service attacks against the Globus Toolkit 4, i.e. the Grid middleware used in our implementation.

The paper is organized as follows. Section 2 analyzes the problem investigated in this paper and the requirements a solution must satisfy. Section 3 presents our approach to secure combined Grid/cluster environments. Section 4 describes implementation issues. In section 5, related work is discussed. Section 6 concludes the paper and outlines areas for future work.

2. PROBLEM STATEMENT

In this paper, we deal with two major problems currently hindering Grid adoption, which we encountered during our

GridNets 2007 October 17-19, 2007, Lyon, France.
 ISBN 978-963-9799-07-3.
DOI 10.4108/gridnets.2007.2160

work on the German national Grid project D-Grid [4]. The aim of the D-Grid project is twofold. In the first phase, a research Grid is to be created linking the existing high performance compute resources of German universities and research institutions in a free academic Grid. The second phase is to encourage a pay-per-use of the Grid by industrial users. The first problem concerns the addition of new attacks against existing cluster system through the introduction of Grid middleware. The second problem refers to new threats against the Grid security infrastructure.

2.1 New Threats Against Clusters

The introduction of a large number of locally unknown Grid users to the local cluster environment is not welcomed by the cluster administrator or by existing cluster users. While standard operating system user protection schemes are in place, the idea of running unknown code and letting unknown users work on the local system is not acceptable. To deal with this concern, we previously introduced a Xen [5] based virtual execution environment into which Grid users are placed so they can not harm other users. Using an Image Creation Station (ICS), Grid users create a Xen image in which they can interactively install their software. This image is then deployed onto the cluster. This counters the concerns of the cluster administrator and the cluster users where the Grid users are concerned, since all Grid users are constrained to their virtual machine and do not have access to the rest of the system [20].

However, a further concern of the cluster administrator and the cluster users is the introduction of relatively young and uncertified Grid middleware into the existing secure local cluster setup. The standard Grid setup of Globus, gLite and Unicore calls for the Grid headnode and the cluster headnode to be on the same physical machine or at least in the same network. This endangers the local cluster and its users, since a remote exploit of the Grid middleware [7, 8, 9, 10] allows an attacker to also compromise the cluster network. Furthermore, the Grid headnode must be reachable from the Internet, exposing the cluster to external attacks, which in a purely local setup would not be possible.

To combat this threat, we propose a Grid enabled dual laned Demilitarized Zone (DMZ) which separates the Grid headnode and the cluster network. However, the creation of a Grid DMZ creates a number of new problems which must be dealt with. Previously, the Grid headnode was responsible for the distribution of job data via direct access to the cluster scheduler. Due to the newly introduced network separation, this approach is no longer possible, since an open connection from the Grid headnode into the cluster network would also open a route for attackers. Furthermore, this separation of the Grid headnode from the cluster creates some security concerns for the Grid users, which leads to the second problem area discussed in the next subsection.

2.2 New Threats Against Grids

Due to the introduction of a DMZ, we face the problem that the Globus Security Infrastructure (GSI), which is the security solution for the Globus Toolkit and gLite, is no longer sufficient to ensure the safety and integrity of user data. In a standard setup, GSI is responsible for the integrity of the data, but the assumption is that the Grid headnode has full access to the cluster. Data encrypted with the GSI is decrypted by the Grid headnode - now in the DMZ - which potentially is compromised. As a consequence, GSI-secured job data stored on the Globus headnode inside the DMZ is no longer safe respectively private. This is unacceptable especially for industrial adoption. Customers wanting to access external Grid cluster resources need to know that their software and data is protected not only during transmission but also during computation on the cluster resources. To this end, we propose an extension of GSI, which cryptographically links the Grid client, through the DMZ and the cluster scheduler, with a virtual execution host, which we introduced in previous work [20, 6], to enable end-to-end cryptographic protection of data.

While the introduction of the DMZ protects the cluster users from Grid attacks and the new encryption scheme protects the safety and integrity of Grid job data, there is still the danger of denial-of-service attacks against insecure Grid middleware in the DMZ. To face this threat, we need a setup that enables us to detect possible attacks and allows us to take appropriate countermeasures. To accomplish this task, we extend standard Network Intrusion Detection Systems (NIDS) by Grid specific attack signatures.

From the discussion above, the following requirements can be derived:

1. To protect existing cluster systems, a separation of the Grid/cluster network into two isolated networks is needed.

2. To protect Grid user data, GSI encryption must be extended to encompass both the Grid and the cluster network.

3. Grid user data must never be stored in unencrypted form in the DMZ, and the Grid middleware must not be able to decrypt the data, since the Grid middleware is in the DMZ and thus is potentially compromised.

4. To protect the Grid middleware in the DMZ, standard Network Intrusion Systems need to be extended by Grid specific signatures.

An important aspect is that the security countermeasures do not add complexity for the end users, since Grids tend to be too complex even without security. All security extensions should therefore be transparent and easy to use.

3. APPROACH

In this section, an end-to-end security solution is presented, which fulfills the stated requirements. The approach is divided into several parts:

3.1 Demilitarized Zone

To prevent external intruders from accessing the cluster hardware by exploiting weaknesses in the Grid middleware, the Grid/cluster subnet is divided into two separate subnets, the border network and the cluster network. The DMZ guards both networks with a firewall configured to the specific needs of the network in question. The border firewall filters connections from the Internet and denies unwanted connections to all machines within the DMZ. However, since Grid middlewares require a large number of open ports to function correctly and efficiently, and a large number of fluctuating users needs to access the Grid, the border firewall is relatively open. The Grid headnode is located in the DMZ.

The inner firewall guards the cluster network and prevents direct connections to the cluster subnet. To protect the cluster network, the inner firewall is very strict and only allows a single specially designed cluster connector to pass through and does not allow any interactive sessions to pass into the cluster network. The cluster resides in the cluster network and consists of a cluster headnode and a set of virtual worker nodes. An architectural overview is presented in figure 1.

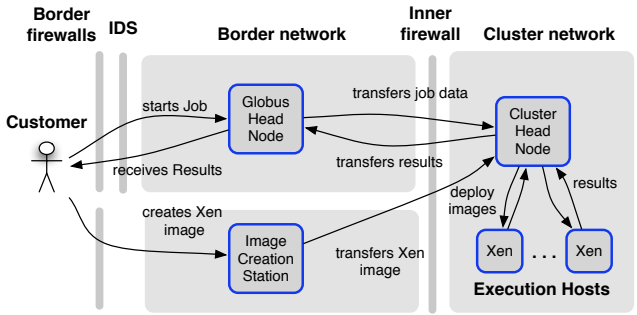

Figure 1: Architectural overview

As mentioned in the previous section, if a user wants to submit a new job, (s)he has to create a Xen image using the ICS. Since the ICS must also allow users to log on from the Internet, it is located inside a part of the DMZ. Based on the X.509 Grid identity, the user can log on to an individually created image and install all required software. The user only gets to log on to his/her own XenU user image and thus can not compromise other images in the DMZ. To prevent an attacker from compromising the Grid headnode and then from there compromising the ICS, a dual laned DMZ approach is employed which restricts access to either the Grid headnode lane or the ICS lane. Thus, if one lane is compromised, the other lane remains unaffected.

3.2 End-to-End Encryption

Since jobs are no longer executed in the same network realm as the Grid headnode, a new encryption scheme is required. When a user submits a job, the client software (e.g. the Gridsphere portal [13]) generates a 48-byte session key which is encrypted with the public key of the XGE (Xen Grid Engine), a modified version of the Sun Grid Engine used on the cluster headnode (see section 6.1). This session key is used to encrypt both the job data and later also the results. Both the job and the encrypted session key can now be transfered through the insecure DMZ. Due to its location in the DMZ, the Grid headnode is considered unsafe, so the job data remains encrypted and the corresponding keys are not available outside of the secure network environment.

3.3 Job Submission, Transfer and Execution

Since direct communication between the Grid headnode and the cluster scheduler is no longer possible, a new mechanism is required to transfer and execute a Grid job. A newly developed software called *Fence* deals with the task of transferring job data to the cluster headnode. During the development, we had to deal with two important issues: Integration into the existing infrastructure (in our case Globus Toolkit 4 and Sun Grid Engine) had to be easy and the software should meet certain security criteria. Due to this, we split up Fence into two parts: One part is a Job Manager and thus responsible for the communication with the Globus Toolkit. The other part is a set of daemons running on the two headnodes and responsible for job data transfers. In detail, Fence consists of the following components:

- Job Manager: It is tightly integrated into the Globus Toolkit. One component is a Scheduler Event Generator (SEG). The other one hands over a new job to the scheduler.
- DMZ Head Node Client (*dhnc*): It provides the communication between the Globus headnode and the Cluster headnode.
- Cluster Head Node Daemon (*chnd*): It represents the interface between the services on the Globus headnode and the XGE on the cluster headnode. Due to its location, the *chnd* is a security critical component and needs additional protection.
- DMZ Head Node Daemon (*dhnd*): This is the second service on the Globus headnode. It is a background task, serving requests from the *chnd*.

To use the developed Job Manager, a new job must use a specific factory type. In the last stage of processing, Globus hands over the job to the Job Manager. The Job Manager extracts all relevant facts from the job description, stores the encrypted job data locally and hands over the job to the next component. The *dhnc* sends a message to the Cluster headnode. The message only contains the ID of the new job. On the cluster headnode, the *chnd* receives the message and, if the job is to be accepted, initiates a connection back to the Globus headnode. This technique allows us to open only one port (for the notification message) in incoming direction on the internal firewall. All following connections are initiated from within the cluster network. The counterpart of the *chnd* is the *dhnd*. These two components exchange the job data over the established connection. After successful transmission, the connection is closed.

The XGE executes jobs not in a native but in a virtualized environment, namely the user specific Xen-Image created in the ICS.

A job submission with full encryption is divided into several steps. The steps are shown in Figure 2 and explained in the following:

1. The user creates a fully customized Xen-Image for execution at the ICS. This image represents the base for the upcoming computation.
2. The image is transfered to the cluster headnode. The connection is initialized from the cluster network. This ensures maximum security, because no open incoming port towards the cluster network needs to be open.
3. The client generates a session key and encrypts the session key with the XGE public key. It then archives and encrypts the job data with the session key and creates a customized RSL file. The RSL file contains the name of the archive and the encrypted session key.
4. The job data is copied to the Globus headnode via GridFTP.
5. A Globus GRAM call according to the RSL file is launched.

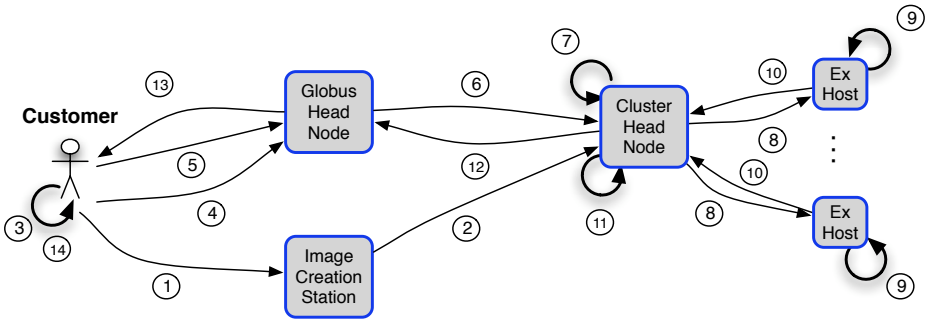

Figure 2: All steps required for an end-to-end encryption

6. Globus hands over the job to Fence, which transfers the new job to the Cluster headnode.

7. The XGE parses the job description, decrypts the session key and decrypts the data.

8. The XGE schedules the images created in step 1.

9. The job is executed.

10. The results are copied back to the XGE.

11. The results get encrypted with the session key.

12. The results are copied back to the Globus headnode.

13. The user fetches the results with GridFTP.

14. The user decrypts the results.

3.4 Intrusion Detection

To detect possible intruders in advance, a NIDS is installed between the border firewall and the border network. Most intrusion detection systems are not able to detect attacks against Grid/cluster environments due to Grid specific attacks and thus new rules have to be developed. There are a number of relevant attacks against a Grid:

- Simple Denial-of-Service Attacks: This type is very similar to common DoS attacks, except that they focus on Grid components, such as the Globus Toolkit, gLite, Unicore, GridFTP, etc.

- Complex Denial-of-Service Attacks: Complex DoS attacks try to misuse certain components of the Grid software. For example, an intruder could generate numerous false certificates to generate a high load on the Grid AAI components.

- Use of exploits: No software component is really free of bugs, so the use of an exploit against a security vulnerability is always possible.

4. IMPLEMENTATION

This section describes the implementation of the DMZ, the GSI extension and the NIDS.

4.1 Demilitarized Zone

The restrictions on the border firewall have to be minimal so as to not impede standard Grid operations. All invalid network traffic (e.g. unrouteable packets, port scans, packets outside of the Grid port range) will be filtered anyway. The Globus Toolkit needs special attention, because it uses a large dynamic port range, and aggressive firewall rules can interrupt normal operations. Further details on Globus and firewalling are discussed in the related work section.

The rules for the inner firewall are more restrictive. Connections initiated from the border network towards the cluster headnode are denied entirely, except for one port needed to transfer job data, which is handled by Fence. Connections towards this port are only allowed from the Globus headnode. Connections from the subnet where the ICS resides are also forbidden. Internet connectivity for the cluster headnode is provided via Network Address Translation (NAT). Both firewalls use stateful inspection to minimize load.

4.2 End-to-End Encryption

The following steps are required for end-to-end encryption.

- Session key generation and job submission: At first, the session key is generated. The software encrypts the job data with this key. Furthermore, the session key is encrypted with the XGE public key and appended to the GRAM-RSL file. Then, a GRAM call is sent to the Globus machine, and GridFTP is used to transfer the encrypted job data.

- New job arrives: After a new job arrives at the Globus headnode, Fence transfers this job to the cluster headnode. All transfered data remains encrypted at all times.

- New job arrives at the cluster headnode: After the XGE recognizes a new job it extracts the session key from the job description and decrypts the job with the private key and starts the corresponding user image. XGE then starts the computation.

- Computation finished: After successful computation, the results are encrypted within the secure virtual environment and are then handed back to the Globus headnode.

To handle the integration of session keys into Globus, we use the RSL extension field and introduce two new tags: `<inputArchive>` and `<ID>`. On the client-side, the job data is

stored inside a Zip-archive (named by the first tag), which is encrypted with the session key. The second tag holds the encrypted key. Encrypting the session key with a public key ensures that we can transfer the key over insecure channels and store the key on untrusted storage e.g. the Grid headnode. The secret key is stored securely on the cluster head node and is not accessible by any users.

4.3 Job Submission, Transfer and Execution

The separation of the cluster network into two networks developed the need for a new software to transfer job data from one headnode to another.

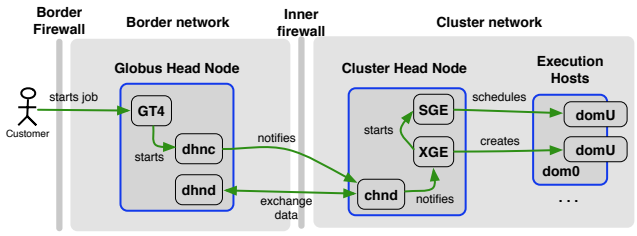

Figure 3: Interaction of Fence and the XGE

Figure 3 displays the interaction of Fence and the XGE. After a customer starts a job via Globus GRAM, Globus starts the *dhnc*. It announces the new job via a message to the *chnd*. To get the related job data, the *chnd* initiates a connection to the *dhnd* from within the cluster network. All data is exchanged over this connection. When all data has been transfered, the *chnd* notifies the XGE about the new job. The XGE starts the user image, transfers the job data into the image and starts the computation. When the computation is finished, the XGE transfers the results back to the Globus headnode. The user can get the results with either GridFTP or SSH.

All components, except for the the Job Manager, are written in C. This enables a high performance and a small memory footprint. Due to the nature of C, a careful source code audit was necessary to avoid the common pitfalls, e.g. buffer overflows or format string vulnerabilities. The two daemons, *chnd* and *dhnd*, need special attention due to their prominent position. Thus, these two daemons run as unprivileged users and in a *chroot* environment. Chroot is an operation on Unix operating systems that changes to root directory. An application inside a chroot "jail" cannot name directories outside this jail. This provides a convenient way to create a sandbox-like environment. Nevertheless, it is possible to escape from such a environment [2]. To minimize this risk, it is recommended to install a kernel hardening patch, e.g. [15] or [24].

4.4 Intrusion Detection

To protect the Grid infrastructure, Snort [23] is placed between the border firewall and the border network as our Network Intrusion Detection System (NIDS). Snort analyzes all incoming traffic and compares it against its signature database. If a positive match occurs, an alarm is raised. To enhance the security of the installed Globus headnode, we extended the signature database with Globus specific attack signatures. Based on the classification in the design section, we developed some proof-of-concept DoS attacks against GridFTP and the Globus container. Based on the evaluation of these attacks, we created new Snort detection rules. One of the rules to detect a DoS-attack against GridFTP is as follows:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET \
2811 (msg:"Globus GridFTP DoS attack"; \
flow:stateless; flags:S; threshold: type \
both, track by_src,count 1000,seconds 30; \
classtype:denial-of-service;)
```

The rules log the message "GridFTP DoS attack", if more than a 1000 TCP packets arrive within 30 seconds. This high thresholds ensures that the normal volume of legal Grid connections observed at our site does not raise a false alarm. The threshold needs to be adjusted to each site's expected usage. Since the attacking packets try to establish a connection, the SYN flag must be set. The classification of the rule is set to DoS.

If an attack is registered, there are three responses: First, a log entry is written, solely for documentation purposes. Second, the Grid and cluster administrators receive an alert message. Third, the attack is displayed on our local WebMDS, which enables end users to view Globus GT4 monitoring information via a standard web browser interface. The icon for the registered resource changes from green to red to enable a quick visual recognition of which sites are under attack.

5. EXPERIMENTAL RESULTS

While the presented features (DMZ, Fence) are required for effective security in Grid/cluster environments, they create a certain amount of overhead. The transmission time is extended due to the extra hop via Fence, and the en- and decryption also consume some time compared to a completely insecured Grid operation.

The test environment consists of 6 machines connected with a 100 Mbit ethernet. The firewalls are Pentium III machines with 1 GHz, 512 MB RAM and FreeBSD installed. The Globus headnode is a Pentium IV with 3 GHz, 1 GB RAM and Debian Linux installed. The same configuration applies to the ICS and the client machine. The Cluster headnode is a Pentium IV with 1.8 GHz, 512 MB RAM and Solaris 10 installed. There are also 4 Pentium IV with 1.8 GHz, 512 MB RAM worker nodes running Debian Linux in a Xen environment.

The NIDS is a passive component, which does not influence job execution time.

The most important measurement concerns end-to-end encryption. It introduces improved security, but it also consumes time. To show that the introduced overhead is negligible, we conducted two measurements. The first one measures the time for an encrypted job from the client to the XGE. The second measurement is the same, except that the data is unencrypted. The execution time is job dependent and thus is factored out of the measurements.

Figure 4 shows the times for the following steps:

1. Start a job on the client side and encrypt the job data with a 48-byte key
2. Transfer the job data via Fence to the XGE
3. The XGE detects the new job and decrypts the data

Our test application is the casting simulation package CASTS (Computer Aided Solidification TechnologieS) [1] which is one of the engineering applications in the D-Grid. CASTS is a dedicated software tool developed for the full range of metal casting processes. CASTS calculates transient temperature distributions in mold, core and alloy, taking into account both latent heat release as a function of fraction solid, and heat transfer resistance at material interfaces. The typical data volume for CASTS is about 290 MB.

We conducted 50 trials to get a robust mean, which is about 154 seconds. The chart is divided into three parts. The bottom part displays the time needed to encrypt the data. The middle part displays the time needed to transfer the files via Fence from the Globus headnode to the Cluster headnode. The upper part displays the time needed to decrypt the data. The difference between the two cryptographic processes results from the diverse hardware (1.8 GHz CPU vs. 3 GHz CPU). The small variance in the transfer time is due the polling nature of the XGE scheduling algorithm.

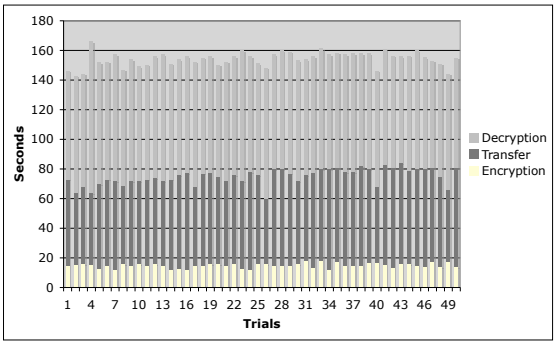

Figure 4: Time for fully encrypted job submissions in 50 trials

The second measurement is exactly the same as the first, except no job data is encrypted respectively decrypted. Figure 5 shows the time in seconds on the ordinate and 50 trials on the abscissa. The mean is about 49 seconds. Due to the watchdog characteristic of the XGE, we can see the same variance here as already described in the last case.

The difference between the means of the two trials is 105 seconds, so we need about twice the time for full end-to-end encryption. Compared with the time this CASTS job needs to complete (about one hour), the extra time (plus 105 seconds) is worth the gained security.

To compare the filter overhead introduced by the firewalls, we transferred data with and without the firewalls. To gain a robust mean, we transferred the data 100 times. The volume of the data is the same as above. Figure 6 shows the results of the measurement. The upper curve displays the transfer time through the firewalls, the lower curve displays the transfer time without firewalls. The mean of the upper curve is about 26.7 seconds, the mean of the lower curve is about 25.7 seconds. The resulting difference is very small, so the measurements indicate that the use of firewalls for a DMZ helps to improve the security for little cost.

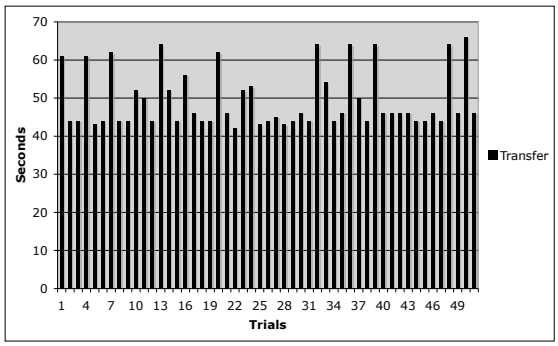

Figure 5: Time for unencrypted job submissions in 50 trials

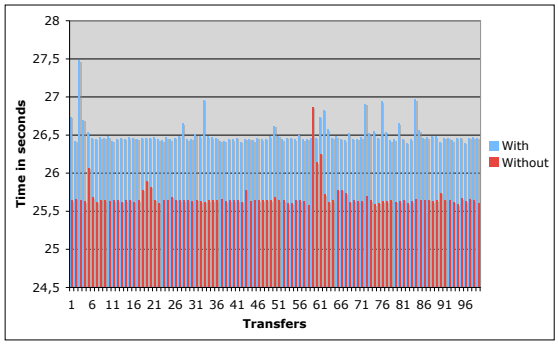

Figure 6: Transfer data chart with/without a firewall

6. RELATED WORK

6.1 Grid DMZ

Connecting the Sun Grid Engine to the Globus Toolkit as a backend scheduler is dealt with in [11]. The software is the current state-of-the-art solution for scheduling Grid jobs to the SGE and is widely used. However, it is not DMZ capable, since it requires direct access from the Globus headnode to the Sun Grid Engine. This can either be achieved by installing the SGE locally on the Globus headnode, or the execution environment must be made accessible via the Network File System or a similar direct access solution. Both integration options are not acceptable from a security perspective, which led to the development of Fence.

The software used on the Cluster headnode is the Xen Grid Engine (XGE), developed by Fallenbeck et. al [6]. The XGE is a modified version of the Sun Grid Engine ([19]), which supports virtualization. A new job is executed inside a personalized Xen-Image, created by the ICS. This ensures the integrity of the user data, since the utilized virtualization techniques create an isolated instance of an operating system to which only the owner has access.

Globus offers a wide range of remote services, so firewalls rules have to be chosen carefully in order to avoid disturbing legal users. Von Welch [25] analyzes Globus versions 3 and 4 in terms of network ports and data streams. Based on that, a fine-grained firewall configuration can be created, so that authorized users can work without disruptions, while at the same time blocking most unwanted traffic. A similar study was done by Baker et al. [3].

6.2 Intrusion Detection Systems

The Globus Toolkit provides the Grid Security Infrastructure (GSI) [22] to guarantee authentication, authorization and integrity of the data in transit and utilizes Unix security to ensure the safety of data on the compute nodes in traditional Grid/cluster setups, where the Grid headnode and the cluster headnode are in the same network. Since the DMZ scenario is not considered, GSI does not offer protection for this setup.

Several publications deal with integrating an Intrusion Detection System into a Grid. Schulter et. al [18] describe a Grid IDS, which combines Host- and Network-IDS to analyze the users' behavior. A scheduler loads the users' profiles and starts one or more analyzer processes to detect anomalies. All components interact closely with a database to update changed profiles regularly. The IDS utilizes stored user behavior to detect anomalous activities. Due to the fact that in our DMZ setup the Grid headnode can not see most of the users' Grid activities, this approach is not applicable to our scenario.

Fang-Yie Lue et. al [12] also integrate an IDS into a Grid. Their solution uses existing Grid resources to detect high volume packets, especially DDoS attacks. Instead of standard technologies, they use their own solution to overcome possible performance bottlenecks. Their approach mainly deals with the distribution of load for the IDS and requires several Globus nodes to be utilized, which for our scenario is not necessary. Furthermore, it is not clear whether an implementation of the system is available.

As a consequence, we do not use the above Grid IDS, but extend a standard NIDS. Snort [23] is the de facto standard for NIDS and widely used within the open source community. This brings with it the benefit of a large user community and a stable code base. The rule extension created as part of this work can easily be integrated into existing Snort setups, thus simplifying adoption.

7. CONCLUSIONS

This paper presented an end-to-end security concept for Grid/cluster environments. A novel dual lane Grid Demilitarized Zone (DMZ) was introduced, securing existing cluster setups from attacks stemming from the open Grid world. The dual lane nature of the DMZ further subdivides the DMZ into sub-compartments, allowing for secure network based image creation and job submission, respectively. The Globus Security Infrastructure (GSI) was extended to cope with the new DMZ and virtualization setup created for the protection of the local cluster. Job data is encrypted during all stages of transmission and storage in insecure networks. The extension of a Network Intrusion Detection System (NIDS) by several Grid specific rules further protects the Grid network. Due to the use of a Xen-enabled Sun Grid Engine called XGE running on the cluster headnode, Grid users are isolated from each other and from cluster users. This novel Grid setup allows both academic and industrial customers to ensure the safety of their data while using external Grid resources and at the same time protects local users from external attacks. The additional overhead created by the security measures is negligible compared to the total execution time of a typical Grid job and the added security benefits the systems offers.

There are several areas of future work. Currently, Fence is used in conjunction with the Globus Toolkit. To make the software available to a wider audience, new interfaces to other Grid middlewares (e.g. Unicore, gLite) have to be developed. In the context of deployment of the setup onto a D-Grid production cluster, further measurements will be made to assess the performance with different job types. Finally, further Grid specific IDS signatures are needed to enhance the installed NIDS as new attacks appear both in the lab and in the wild.

8. ACKNOWLEDGMENTS

This work is partly supported by the German Ministry of Education and Research (BMBF) (D-Grid Initiative).

9. REFERENCES

[1] Access e.V., Aachen. CASTS Homepage. http://www.access.rwth-aachen.de/00010_00013_process_simulation.htm. May 2007.

[2] Anton Chuvakin. *Using Chroot Securely.* http://www.linuxsecurity.com/content/view/117632/49/. May 2007.

[3] Mark Baker, Hong Ong and Garry Smith: *A Report on Experiences Operating the Globus Toolkit through a Firewall.* Version 1, Distributed Systems Group, University of Portsmouth, September 2001.

[4] D-Grid Initiative. *D-Grid Integrationsprojekt.* http://www.d-grid.de/. May 2007.

[5] B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, I. Pratt, A. Warfield, P. Barham, and R. Neugebauer: *Xen and the Art of Virtualization*, Proc. of the ACM Symposium on Operating Systems Principles, 2003.

[6] Niels Fallenbeck, Hans-Joachim Picht, Matthew Smith and Bernd Freisleben: *Xen and the Art of Cluster Scheduling.* Super Computing 06, Virtualization Workshop, IEEE, 2006, pp. 237-244.

[7] Globus Security Team. *Globus Security Advisory 2007-03: Nexus Vulnerability.* http://www.globus.org/mail_archive/security-announce/ 2007/05/msg00000.html, May 2007.

[8] Globus Security Team. *Globus Security Advisory 2007-02: GSI- OpenSSH Vulnerability.* http://www-unix.globus.org/mail_archive/security-announce/2007/04/msg00000.html. March 2007.

[9] The Grid Security Vulnerability Group. *Critical Vulnerability: OpenPBS/Torque.* http://security.fnal.gov/CriticalVuln/openpbs-10-23-2006.html. October 2006.

[10] Internet Security Systems. *UNICORE Client Keystore Information Disclosure.* http://xforce.iss.net/xforce/xfdb/30157. November 2006.

[11] The London eScience Centre: *Sun Grid Engine Integration with Globus Toolkit 4*. http://www.lesc.ic.ac.uk/pro jects/SGE-GT4.html, February 2007.

[12] Fang-Yie Leu, Jia-Chun Lin, Ming-Chang Li, Chao-Tung Yang and Po-Chi Shih: *Integrating Grid with Intrusion Detection*. AINA05: Proceedings of the 19th Int. Conference on Advanced Information Networking and Applications, pp. 304-309, Washington, DC, USA, 2005, IEEE Press

[13] The GridSphere Developers. *The GridSphere Portal Framework*. http://www.gridsphere.org, May 2007.

[14] Syed Naqvi, Michel Riguidel. *Threat Model for Grid Security Services*. In Proc. European Grid Conference (EGC 2005), pp. 1048-1055, Amsterdam, The Netherlands, 2005.

[15] The Openwall Project. *Linux Kernel Patch From The Openwall Project*. http://www.openwall.com/linux/. May 2007.

[16] Prelude Development Team. *Prelude Hybrid IDS project*. http://www.prelude-ids.org/, February 2007.

[17] A.L. Rowland, M. Burns, J.V. Hajnal, D. Rueckert D.L.G. Hill: *Using Grid Services From Behind A Firewall*. Imperial College London, 2005.

[18] Alexandre Schulter, Fabio Navarro, Fernando Koch Carlos Becker Westphall: *Towards Grid-based Intrusion Detection*. Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP 2006. Pages: 1-4.

[19] Sun Grid Engine Developers. *Sun Grid Engine Website*. http://gridengine.sunsource.net. March 2007.

[20] Matthew Smith, Thomas Friese, Michael Engel, Bernd Freisleben. *Countering Security Threats in Service-Oriented On-Demand Grid Computing Using Sandboxing and Trusted Computing Techniques*. Journal of Parallel and Distributed Computing, Volume 66, Issue 9, pp. 1189-1204, Elsevier, 2006.

[21] Matthew Smith, Thomas Friese, Michael Engel, Bernd Freisleben, G. Koenig, W. Yurcik:. *Security Issues in On-Demand Grid and Cluster Computing*. Sixth IEEE International Symposium on Cluster Computing and the Grid Workshops (CCGRIDW'06), pp. 24-38, IEEE Press, 2006.

[22] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke. *A Security Architecture for Computational Grids*. Proc. 5th ACM Conference on Computer and Communications Security Conference, pp. 83-92, 1998.

[23] Snort Development Team. *Snort Network Intrusion Detection*. http://www.snort.org, February 2007.

[24] Brad Spengler. *grsecurity*. http://www.grsecurity.org/. May 2007.

[25] Von Welch. *Globus Toolkit Firewall Requirements*. http://www.globus.org/toolkit/security/firewalls/Globus-Firewall- Requirements-9.pdf, October 2006.