

CONTRACT NUMBER 508830

DEISA
**DISTRIBUTED EUROPEAN INFRASTRUCTURE FOR
SUPERCOMPUTING APPLICATIONS**

European Community Sixth Framework Programme
RESEARCH INFRASTRUCTURES
Integrated Infrastructure Initiative

Performance and Security of the general distributed file system AFS.
Deliverable ID: D-SA2-2B

Due date: April, 30, 2005
Actual delivery date: May 15, 2005
Lead contractor for this deliverable: RZG, Germany

Project start date : May 1st, 2004
Duration: 5 years

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	
PP	Restricted to other programme participants (including the Commission Services)	X
RE	Restricted to a group specified by the consortium (including the Commission	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Table of Content

Project and Deliverable Information Sheet.....	1
Document Control Sheet	1
Document Status Sheet.....	1
Document Keywords and Abstract.....	1
Table of Content	2
List of Figures.....	3
1. Introduction.....	4
1.1 Executive Summary.....	4
1.2 References and Applicable Documents	4
1.3 Document Amendment Procedure	4
1.4 List of Acronyms and Abbreviations	4
2. Description and configuration	6
2.1 Introduction	6
2.2 Configuration	6
3. Security.....	9
4. Performance.....	12
5. Other Progress	14
6. Ongoing Investigations.....	15

List of Figures

- Figure 1..... Accessibility of the AFS-cell deisa.org within DEISA
- Figure 2..... Accessibility of the AFS-cell deisa.org from outside DEISA
- Figure 3..... Simple, schematic overview over Kerberos model
- Figure 4..... Read/Write Performance of AFS
- Figure 5..... Credential Transfer in a batch environment

List of Tables

- Table 1 Characteristics of AFS and MC -GPFS

1. Introduction

1.1 Executive Summary

The Service Activity 2 within the DEISA project deals with the connectivity of all DEISA sites on the file system level. Two strategies are pursued in parallel:

- ? Deploying IBM's GPFS. See deliverable D-SA2-2A.
- ? Implementing a distributed file system structure for heterogeneous environment.

This document discusses the advances of the AFS implementation at DEISA since the last Deliverable.

1.2 References and Applicable Documents

- [1] <http://www.deisa.org>
- [2] Deliverable D-SA2-2A
- [3] Acronyms and Abbreviations: <http://cgi.snafu.de/ohei/user-cgi-bin/veramain-e.cgi>
- [4] www.iozone.org, used version: v3.217
- [5] Andrei Masslennikov, CASPUR, "New results from CASPUR Storage Lab", <http://hepwww.rl.ac.uk/hepix/nesc/maslennikov2.ppt>
- [6] <http://www.pdc.kth.se/kth-krb/>

1.3 Document Amendment Procedure

Not applicable.

1.4 List of Acronyms and Abbreviations

AFS	Andrew File System , used in the open-source implementation OpenAFS
AIX	Advanced Interactive eXecutive (IBM's derivative of UNIX OS)
Altix	Multi-Processor compute node from SGI
Apple	Computer Vendor
CASPUR	Consorzio interuniversitario per le applicazioni di supercalcolo per università e ricerca.
CERN	Centre Européen pour la Recherche Nucleaire
DEISA	Distributed European Infrastructure For Supercomputing Applications
FS	File System
GSSAPI	Generic Security Services Application Programmers Interface
GPFS	General Parallel File System, proprietary FS from IBM
HTTP	Hyper Text Transport Protocol, the base protocol of the internet.
IBM	Computer Manufacturer
IP	Inter-Protocol, basic network protocol.

HPS	High Performance Switch
KERBEROS	Security Infrastructure within untrusted networks
Linux	Free, open source UNIX clone.
LPAR	Logical PARTition (subset of a larger system)
MAC-OS	OS used on Apple Macintosh computers
WINDOWS	OS from Microsoft
NFS	Network File System, a very common filesystem.
NREN	National Research and Education Network
OS	Operating System
P655, P690	High performance computing nodes built by IBM.
RPC	Remote Procedure Protocol
RFC	"Request For Comment", technical standards papers
RX	RPC protocol based on UDP
SCSI	Small Computer Systems Interface
SGI	Computer Vendor
SOLARIS	OS from SUN
SUN	Computer Vendor
T10	Protocol within SCSI
TCP	Transmission Control Protocol
UNIX	Operating system family
USB	Universal Serial Bus
UDP	User Datagram Protocol

2. Description and configuration

2.1 Introduction

Common file systems provide one of the foundations of the integrated DEISA environment. The idea behind having a common file system is that deeper integration of the DEISA sites will make it easier for developers to write upper layer software spanning the whole DEISA infrastructure.

The expected functionality such as easy installation, usage, reliability and transparency to the user has been proven by AFS in the last Deliverable D-SA2-1B.

AFS is installed and used in a test-environment between three of the four core-sites (CINECA, IDRIS, RZG) and is used for common tasks such as distributing software.

2.2 Configuration

DEISA environment

The AFS-cell "deisa.org" is up and running in production since May 2004. It can be accessed from all AFS-clients inside and outside the dedicated DEISA network. On the DEISA test machines of three of the four DEISA core-sites the AFS client is installed and AFS is visible.

AFS is used for three tasks: to provide users some distributed space, to share software such as libraries between the sites and to share site-specific data, like list of a nodes of one site connected to the DEISA-network with other sites.

In the AFS-tree under "/afs/deisa.org" the directory "u" contains the mount points of the AFS-home volumes of all DEISA-users. This space can be accessed by all DEISA users from all over the world, regardless if they are inside or outside the DEISA network.

The directory "/afs/deisa.org/@sys" is used for software distribution. The term "@sys" is translated by the AFS-client to the appropriate system architecture and operating system (under AIX 5.2 this is "rs_aix52"). Architecture and operating specific software and libraries are stored under this directory in the subdirectories "bin" and "lib". Like this, a user just needs to add "/afs/deisa.org/@sys" to its PATH variable and automatically gets the right binaries for his/her architecture. The same is true for shared libraries. This means, library and executable access is transparent to the user not only across all participating AFS-sites but also across hardware-architectures.

Then there are directories for each DEISA site where site-specific information or configuration files can be stored. At present 2.5 TB of disk space are available for the AFS cell "deisa.org".

RZG creates automatically for each new DEISA user a principal in Kerberos 5 and an AFS home volume in the cell "deisa.org".

Figure 1 shows the server-client configuration within the DEISA-network. The only server running currently is located at RZG, but installing more servers at the other sites is a trivial task. Thus, the AFS-client machines at RZG have purely local access whereas

those of IDRIS and CINECA have remote access only. This is subject to change when file servers are installed at those two sites. These file servers will have replicas of the software-volumes, so that the access of software and libraries as described above is always local.

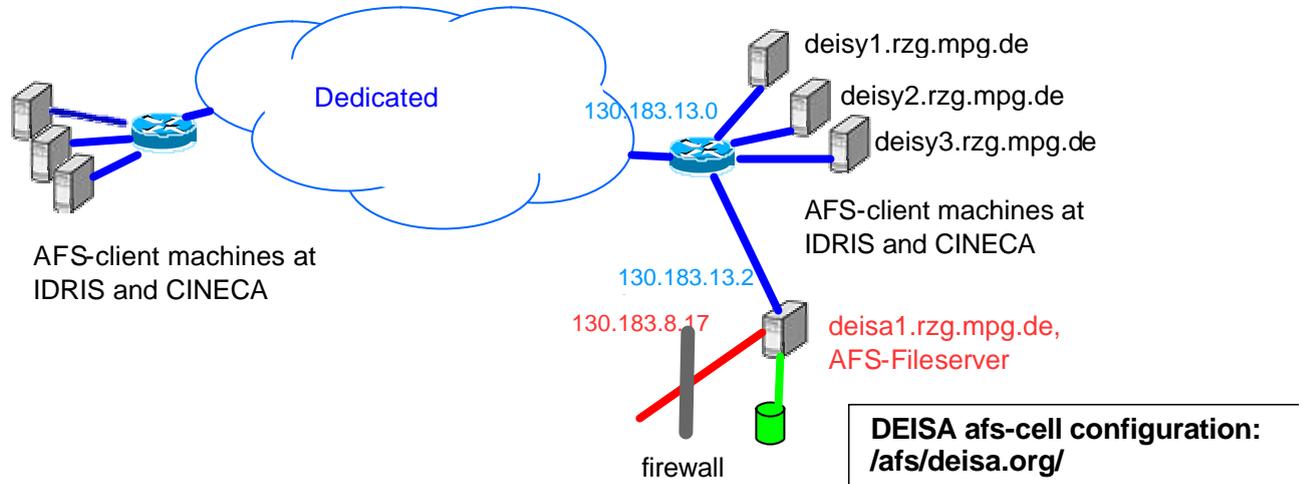


Figure 1: Accessibility of the AFS-cell deisa.org within DEISA

World Wide Access

Scientists are very mobile these days, so it is necessary for them to check the progress of their computations from all over the world. Since one of the DEISA-AFS servers has beside its restricted DEISA-network interface a public one. Figure 2 shows a schematic view of the situation. This does not imply a security problem for the DEISA network, since no routing is done and the AFS-file space has a strong protection (see next section). The benefit of this is enormous: the scientists can from all over the world access the output files of jobs as long as they are in the DEISA-AFS. The only prerequisites are an installed AFS-client on the workstation and proper configuration of the non-DEISA site's firewall. The OS of this workstation is of minor importance. AFS clients exist for about 20 flavours of UNIX as well as MS-Windows and MAC OS.

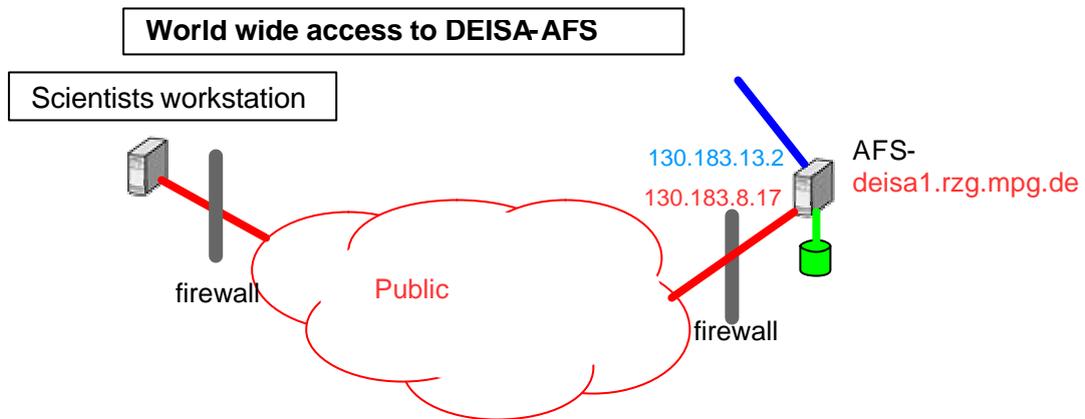


Figure 2: Accessibility of the AFS-cell deisa.org from outside DEISA

3. Security

Files in AFS are protected on the directory-level by a Kerberos mechanism [6] in the version 4, although Kerberos Servers of version 5 can be used. The Kerberos mechanism provides strong security. It is based on the principle of a trusted third party, in this case, the Kerberos servers. It is noteworthy that the security-model of Kerberos is significantly different from that of certificates.

Figure 3 illustrates the security model of Kerberos in a simplified manner.

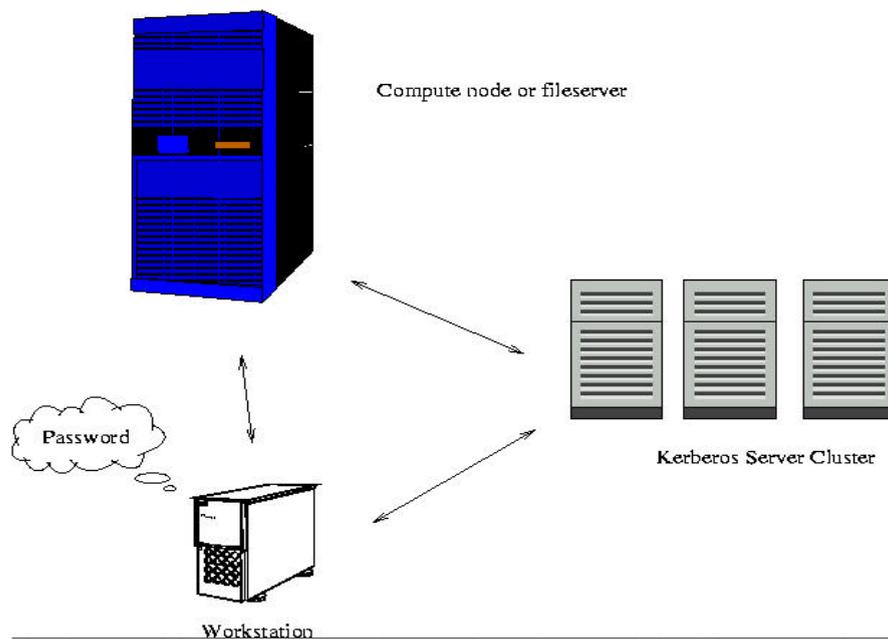


Figure 3: In the world of Kerberos, the workstation and the fileserver do not trust each other initially, they verify with the help of the Kerberos Server the authenticity of the other host.

The compute nodes and client workstations do not trust each other, but only the Kerberos server. Thus, before gaining any rights within the AFS-filespace, the user or a process has to authenticate itself to one of the so-called Kerberos servers.

The Kerberos Server returns the process or user a so-called "credential", a Kerberos ticket which is reduced to an "AFS-token". This "token" exists only in the Kernel-memory of the machine. An AFS token is usually acquired by an interactive authentication. The respective command prompts the user for his or her password and asks one of the Kerberos Servers to grant a token. A token is not indefinitely valid, for interactive use, it has to be renewed after 24h. This minimizes the possibility that if a user forgets an authenticated program on a public computer, somebody else can exploit this. For long running batch jobs, the expiry time is prolonged.

Here lies one of the fundamental differences to the certificate model. In Kerberos, there should be no permanent credential stored anywhere, except on the Kerberos Servers, of course. The user has to type in a password, which is stored only in the mind of the person and gets a time-limited ticket for it. Using certificates, the user has to store his private part on a long-time storage, e.g. a hard-drive or an USB-Stick or something

similar. However, in Kerberos one can also generate so-called key-files which can be used to create tickets, so being similar to certificates.
More detailed information can be found under [5] or similar web-pages.

Future Investigations

Native Kerberos 5 support

With the next OpenAFS version 1.4, Kerberos version 5 will be supported natively. At present, AFS already works with Kerberos 5 servers, but uses the version 4. Kerberos 5 has a couple of advantages over Kerberos 4 like "preauthentication" which render brute-force attacks impossible.

Access to AFS by certificates

As already mentioned above, the mechanism of obtaining an AFS-token in the batch environment has to be changed to gain further security. Once certificates are accessible to all DEISA-users, a mechanism based on Generic Security Services Application Programmers Interface (GSSAPI) should be in place to generate a token from the certificate issued together from the queuing software. A schematic overview over the generation of AFS-Tokens is given in the figure below.

The user signs a token-request, which may be a small plain text file, with the private key of his certificate. This "signed Token-Request" will be submitted together with the job description to the batch-queue master. Once the compute nodes are allocated, the signed Token-Request is passed to the nodes which in turn send it to the Kerberos Server. The Kerberos Server validates the signed Token-Request and determines the user belonging to the certificate used to sign the request. If everything is correct, the Kerberos-Server generates a token and sends it back to the compute-node which then continues starting the job. Since other Service Activities are already working on the implementation of certificate-based authorization, every DEISA user will have a certificate. When this work is stabilised, the GSSAPI mechanism for AFS will be implemented. GSSAPI is to appear in the next official OpenAFS major version 2.0. Then the preferred way will be to pass the user certificate on to batch-queue server which in turn will forward it to the compute node. This one will use GSSAPI to create an AFS-Token which then allows the job-processes to access the users file or licensed software within AFS.

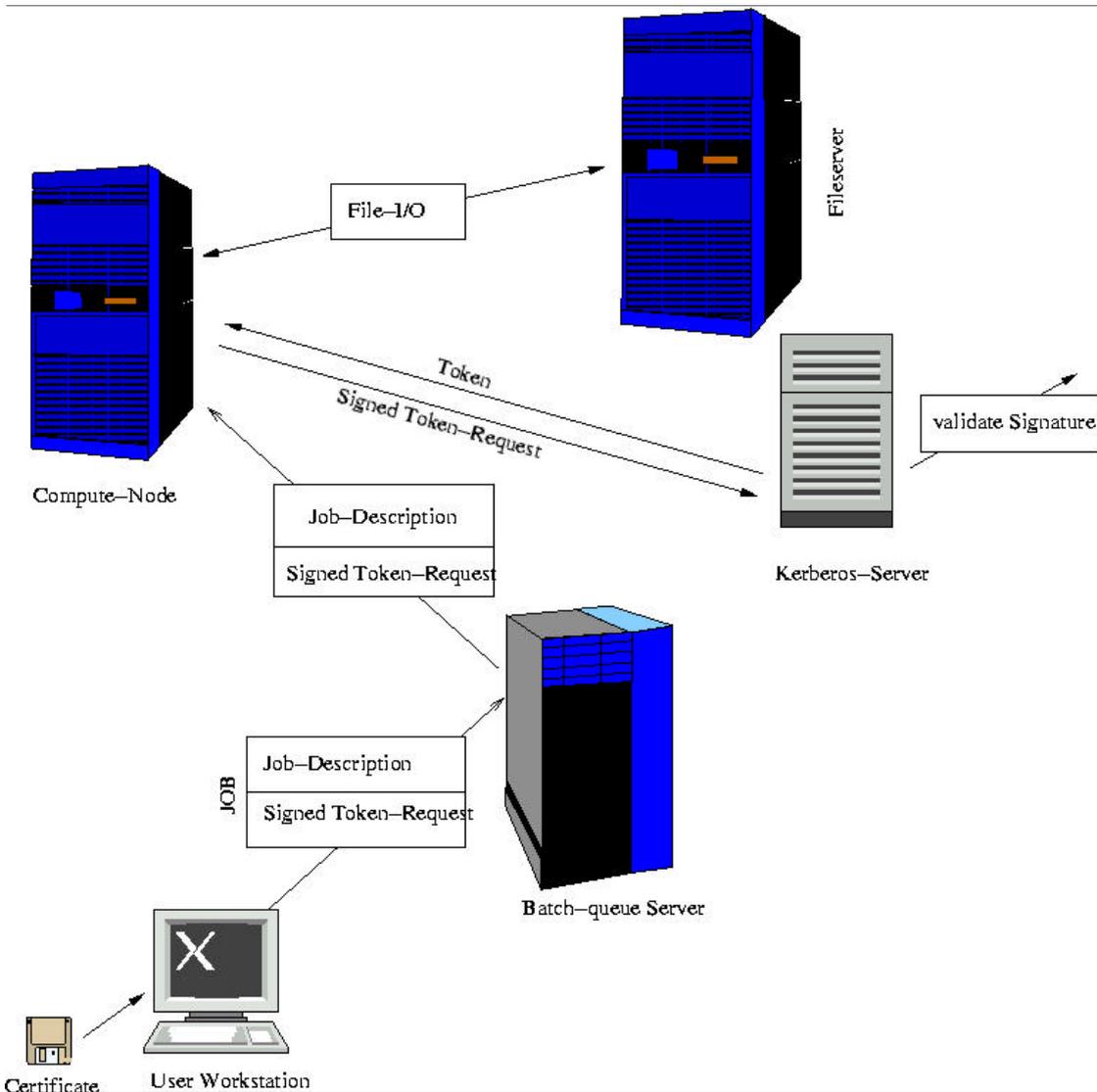


Figure 5: Schematic view of the use of certificates within a batch-environment to allow the compute-node to access the private files of the user while running his/her job.

4. Performance

Improvements for AFS - Improvements of the RX-protocol

The current read and write performance to AFS from remote DEISA sites cannot really take advantage of the band-width of the dedicated network. The UDP-based RX-protocol faces the same problems as TCP: the rather long round-trip time of about 20 ms requires very large windows to reach reasonable performance. While the standard windows size in RX was limited to 32 UDP-packages, RZG and also CERN are making tests to increase the window size. First tests between CINECA and RZG showed that with a windows size of 128 UDP-packages a throughput of some 28 MB/s using 4 AFS-clients is reachable.

Figure 4 shows the three different test-cases:

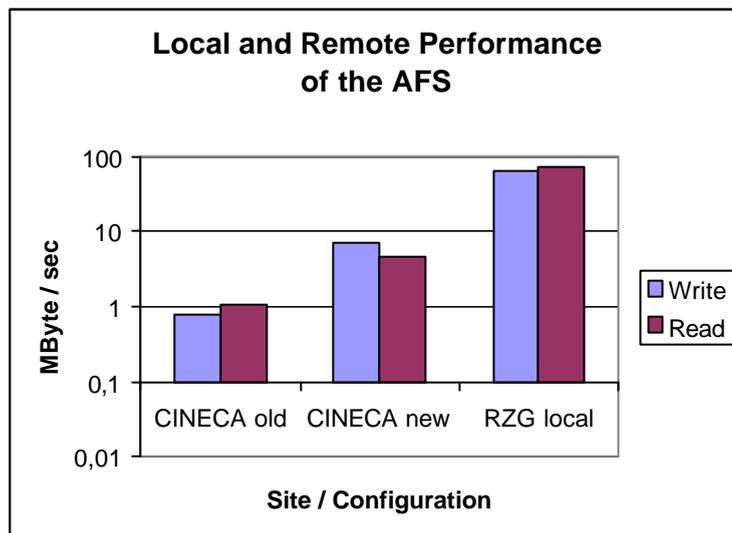


Figure 4: Performance values of a single AFS-client to a single AFS-Server at RZG, once from CINECA in two different configurations and locally at RZG . Note the logarithmic scale, the local performance is at ca. 70 MByte/sec for read and write.

- ? CINECA old: the standard AFS-client
- ? CINECA new : improved version
- ? RZG local: improved version, but no WAN-latency.

When starting the test with the old client at CINECA, there seemed to be some resent packages which – as in the case of TCP – result in a dropping down the window size to rather low values. These resends were not exactly understood. It is now believed that these resends were caused by the faulty firmware of the router at RZG, a more detailed description of this problem is given in the deliverable D-SA2-2A. Thus, the performance improvement between “CINECA old” and “CINECA new” is based not only on the software development mentioned above, but also on debugging of the hardware network-layer. “RZG local” shows finally the local performance of a single client to a

single AFS-server, which saturates about 50% of the theoretical bandwidth and 70% of the measured raw TCP bandwidth. Generally there is no reason why RX should be slower than TCP since its implementation is based on exactly the same RFCs. In real production with more than one data-stream it is therefore expected to make use of the full network bandwidth.

5. Relationship to MC-GPFS

Since two distributed file systems are used within DEISA, the interaction of the two GFSs has to be well defined. The different roles of AFS and MC -GPFS are determined by their respective features (table 1), the user demands and administrators wishes.

	AFS	MC-GPFS
Supported Platforms	many types of Unix, MS-Windows, MAC OS	IBM AIX, SGI Altix
License	OpenSource / proprietary	proprietary
Performance	some MByte/sec on WAN	tens of MByte/sec on WAN
Accessibility	world-wide and DEISA internal	DEISA internal only
Replication	intrinsic	manually
Age	>15 years	0.5 years

Table 1: Characteristics of AFS and MC-GPFS

By deploying the respective advantages of the two file systems, the maximum possible convenience and flexibility is offered to the users.

Possible scenarios include:

- ? File access: It is clear that grand-challenge jobs always use the MC-GPFS as their primary file systems. However, users can take advantage of the world-wide accessibility of AFS. They can from home or when on a journey work on papers and result-files related with jobs running at DEISA, configuration files for the next job or check status information of the currently running job.
- ? Software distribution: Due to the intrinsic replication feature of AFS, it is easier for the administrators, to keep shared software synchronous.
- ? Fallback: The AFS serves for administrators during and after the implementation phase of MC-GPFS, as a fallback solution. The configuration and tuning of MC-GPFS caused many outages. AFS, on the other hand had an outage time of 0.1% in the last year.
- ? Platforms: MC-GPFS and AFS supports most major platforms deployed within DEISA. Although AFS supports more platforms, this is only for the users of interest, but not for the infrastructure serving grand challenge jobs.
- ? License: MC-GPFS is proprietary only, AFS can be used in a proprietary packet from IBM or as a open source solution. This has little effect on the usage of the two GFSs.

6. Additional Progress

Upgrade of the AFS-client to the new OpenAFS release 1.3.79

Much time has been spent to trace a bug in the actual version 1.3.79 of OpenAFS which crashed AIX 5.2 systems. Finally the bug was found and reported to OpenAFS.org so that now the AFS-clients on the DEISA machines can benefit from recent AFS improvements.

AFS in heterogeneous DEISA environments

RZG is testing the OpenAFS client on a SGI Altix machine. This will allow SARA and the future DEISA site LRZ to access AFS on their machines. The DEISA users of LRZ are already accessing the AFS cell "deisa.org" from their Linux and Solaris machines.

Access to AFS from batch jobs

In the batch environment is not possible to obtain interactively an AFS-token. Therefore the token must be bundled with the batch job, as the "LoadLeveler" in the RZG environment is able to do). Alternatively, another authentication mechanism such as certificates must be used to obtain the AFS-token.

In the current pre-production situation both these solutions are not yet in place. Therefore a special technique has been developed to create key-files on the trusted DEISA nodes which are used by a special program to provide the user with an AFS token during login. These key-files are stored locally on each DEISA batch machine in the directory "/var/kerberos" and are protected by ownership and mode-bits in a way that only processes belonging to the DEISA user can access them. If the user changes his Kerberos password his key-file on each DEISA batch machine must be changed accordingly. This job is done with a delay of not more than half an hour by a pair of daemons one of writes an encrypted file into an AFS directory belonging a special AFS user while the other one running on the DEISA batch machine extracts information from this file and updates the key-files.

Though this technique offers exactly the same security for AFS on the DEISA batch machines as for GPFS or local files systems (because the "root" user on this machine can get access to any user's data) it should stopped as soon as the token-transfer and/or acquisition of tokens by means of certificates are in place. Since then, even root cannot access user data in the AFS-filespace.

7. Supplementary ongoing Investigations

Using object storage with AFS

One main difference to GPFS is that AFS files are delivered always by a single file-server, while GPFS files typically are striped over multiple block-I/O servers. Especially within wide area networks, multiple connections allow to gain higher throughput than a single one.

In a common project with CASPUR in Rome and CERN in Geneva, RZG is working on a object storage solution for AFS. The idea is to store in the AFS fileserver only the metadata which are necessary to access a file in the object storage. An AFS client would then be notified that the data of a file must be fetched from one or more object storage servers and the fileserver would provide the client with the encrypted access information which only the object storage servers can decode. This technique is far more secure than block-I/O (used by GPFS) because the access control takes place in the AFS fileserver rather than on the client. Also the danger that a single misconfigured or malicious client could damage a shared file system is excluded because the client has no block level access to the file-system. This point becomes increasingly important with the growing number of machines with access to the file-system.

The access to the Object Storage System could either follow the T10 SCSI standard or be using the RX-protocol as well. The first solution would prepare the use of autonomous object storage disk systems which might appear in the future while the second solution has the advantage to require much less development on the client side whereas also on the server side existing building blocks could be used.

A first proof of concept test has been performed by Rainer Toebbicke from CERN during the CASPUR Storage Lab campaign in spring 2004 [5]. Utilizing a very simple TCP-based object storage it was shown that the new technique can exploit the full Gbit-ethernet band-width.

AFS on top of GPFS

Another approach being tested at RZG is to use GPFS as the file-system where the AFS fileserver stores the files. With a modified OpenAFS client the fileserver would provide the client during open processing of a file with the information necessary to access the file directly. Thus AFS-clients on machines with access to this GPFS file system could make use of the higher GPFS performance when accessing files from this AFS-fileserver while all other clients could access the files by means of the traditional RX-protocol. The advantage over using GPFS natively would be that the files could be accessed from any AFS client all over the world in a secure Kerberos based manner.

Tests in the "deisa.org" cell at RZG have shown in 2004 that the native GPFS data rates can be reached for large AFS files residing in the GPFS filesystem.