



Adapting Open Source tools to cater for a diverse computing infrastructure

Adam Horwich

March 2010

RAL-TR-2010-014

© Science and Technology Facilities Council

Enquires about copyright, reproduction and requests for additional copies of this report should be addressed to:

Library and Information Services
SFTC Rutherford Appleton Laboratory
Harwell Science and Innovation Campus
Didcot
OX11 0QX
UK
Tel: +44 (0)1235 445384
Fax: +44(0)1235 446403
Email: library@rl.ac.uk

The STFC ePublication archive (epubs), recording the scientific output of the Chilbolton, Daresbury, and Rutherford Appleton Laboratories is available online at:
<http://epubs.cclrc.ac.uk/>

Neither the Council nor the Laboratory accept any responsibility for loss or damage arising from the use of information contained in any of their reports or in any communication about their tests or investigation

Adapting Open Source tools to cater for a diverse computing infrastructure

Abstract

The Scientific Computing Technology Group (SCT) computing infrastructure consists of many different functional services, ranging from high performance compute clusters such as SCARF (a facility run for all of STFC) and the National Service for Computational Chemistry Software (NSCCS), to e-Infrastructure like the UK National Grid Service (NGS), and VMware ESX virtualisation technologies. Alongside site wide STFC policies on security, SCT must also provision mechanisms to effectively manage and monitor the security of these services, seamlessly, with a consistent and cohesive approach. Open Source tools represent a cost effective opportunity to manage infrastructure but often prove inflexible over a heterogeneous collection of services. Additionally, where possible, SCT opt to deploy applications that are included as part of the Red Hat Enterprise Linux (RHEL) suite, as these are actively maintained and patched by Red Hat; this is primarily because all systems managed by SCT utilise RHEL or its Scientific Linux variant. This paper will focus on the efforts made to customise and enhance three Open Source tools: AIDE, Pakiti and Syslog-NG and how these changes have benefitted SCT while still remaining manageable and maintainable.

Table of Contents

1. Introduction.....	1
Applications	1
2. AIDE.....	2
Configuring AIDE.....	2
Turning AIDE into a Service	2
Deploying AIDES	4
The AIDES Frontend.....	4
Limitations and Future Development.....	6
Effectiveness	6
3. Pakiti.....	7
Pakiti Reporting	7
Maintaining Legacy Pakiti 1.0.1	7
Effectiveness	8
4. Syslog-NG	9
Development Proposal.....	9
SCT Syslog-NG Mechanism.....	10
Syslog-NG Frontend	10
Effectiveness	11
5. Conclusions	12
6. References	13

1. Introduction

The SCT group within e-Science at STFC has an aim to both develop and support the UK's e-Infrastructure (such as NGS and EGEE) and enable the STFC Facilities computational infrastructure to take best advantage of this. To ensure that these services and systems are fit for purpose and secure, a series of open source tools are employed. Some tools such as Nagios and Ganglia are used natively, or with little modification; however others like AIDE, Pakiti, and Syslog-NG have required some bespoke customisation to provide a consistent view of our infrastructure. These changes are necessary to support the diverse infrastructure SCT manage and to provide functionality normally reserved for commercial products, at minimal effort.

Applications

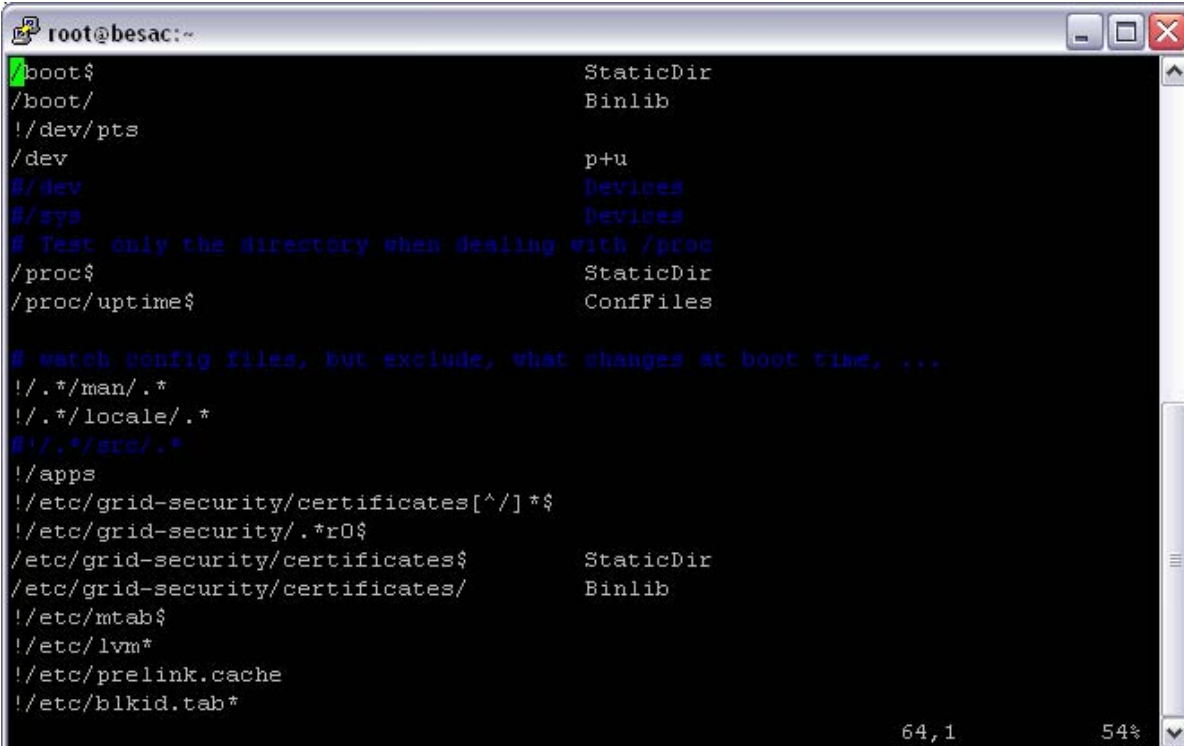
The three applications outlined in this paper have been developed, in most cases, into services that do not change any core functionality of the software. They have been designed to be deployable on 32bit and 64bit Red Hat Enterprise Linux based systems and can deliver the requirements set by SCT for effective system management. They incorporate added resilience and redundancy over their core functionality and have a minimal maintenance overhead. The three applications were also developed in mind to offer future development opportunities.

2. AIDE

The Advanced Intrusion Detection Environment (AIDE) [1] is an application developed by Rami Lehti, Pablo Virolainen and Richard van den Berg to provide a free open source alternative to a product called Tripwire. It is used to verify the integrity of a set of predefined files by generating a known good instance of a file system as a database file that can be compared against the current running system. Since the release of Red Hat Enterprise Linux 4.8, AIDE is offered as a supported application, which proved a decisive advantage when evaluating other products available; Red Hat Enterprise Linux 5 has offered AIDE as a standard application since its release. Installation is provided by an RPM file which will pull in necessary dependencies on a system to ensure the application can function correctly, and also guarantee that it is installed to the correct CPU architecture.

Configuring AIDE

AIDE uses a configuration file that generates rules for file system scans.



```
root@besac:~  
#boot$ StaticDir  
/boot/ Binlib  
!/dev/pts  
/dev p+u  
#/dev Devices  
#/sys Devices  
# Test only the directory when dealing with /proc  
/proc$ StaticDir  
/proc/uptime$ ConfFiles  
  
# warn config files, but exclude, what changes at boot time, ...  
!/*.*/man/*. *  
!/*.*/locale/*. *  
#!/*.*/src/*. *  
!/apps  
!/etc/grid-security/certificates[^/]*$  
!/etc/grid-security/*. *r0$  
/etc/grid-security/certificates$ StaticDir  
/etc/grid-security/certificates/ Binlib  
!/etc/mstab$  
!/etc/lvm*  
!/etc/prelink.cache  
!/etc/blkid.tab*
```

Figure 1.0

These rules, shown in **Figure 1.0** are similar to regular expressions on files and directories which should be scanned, those which should be skipped and explicit rules on exact matches. SCT maintains a single configuration file for all systems scanned, meaning that it will include rules for directories and files which do not exist on some systems. This is handled within AIDE and does not cause any complications other than a notice about missing files. The list of directories to scan is not exhaustive, as this increases complexities and scan time. Instead the list has been tailored to address critical files and components of the RHEL operating system, and the additional files SCT augment systems with during creation.

Turning AIDE into a Service

AIDE by itself did not prove a particularly useful application when attempting to manage a large infrastructure, as its default mode of operation was to run on a client system, comparing its database snapshot with the current file system. When monitoring high performance computing nodes, this would have proved an unacceptable overhead and increased complexity when producing a distributed service.

The solution instead was to rely on a mode of operation which compared two database files generated from the same system. In this mechanism, a known good 'database' is kept in a secure location, and daily snapshots are uploaded to a central AIDE server to be compared for changes. As data is only pushed in one direction, any developments to making a distributed service would be greatly simplified. This did however produce a new set of challenges, incorporating security mechanisms and integrity of scans, looking at validation for false positives and reporting useful information to system administrators. To overcome the problem of uploading data securely to a server, an existing mechanism developed for the SCT password cracking service was leveraged, that uses a secure copy (SCP) method allowing only a server defined white-list of files to be uploaded and to a specific location based on the client submitting the data and when. Clients also take a scan of /proc/uptime to keep a reference of when the scan was performed in an attempt to add authenticity to the scan, a list of recently installed RPMs and basic administrator information.

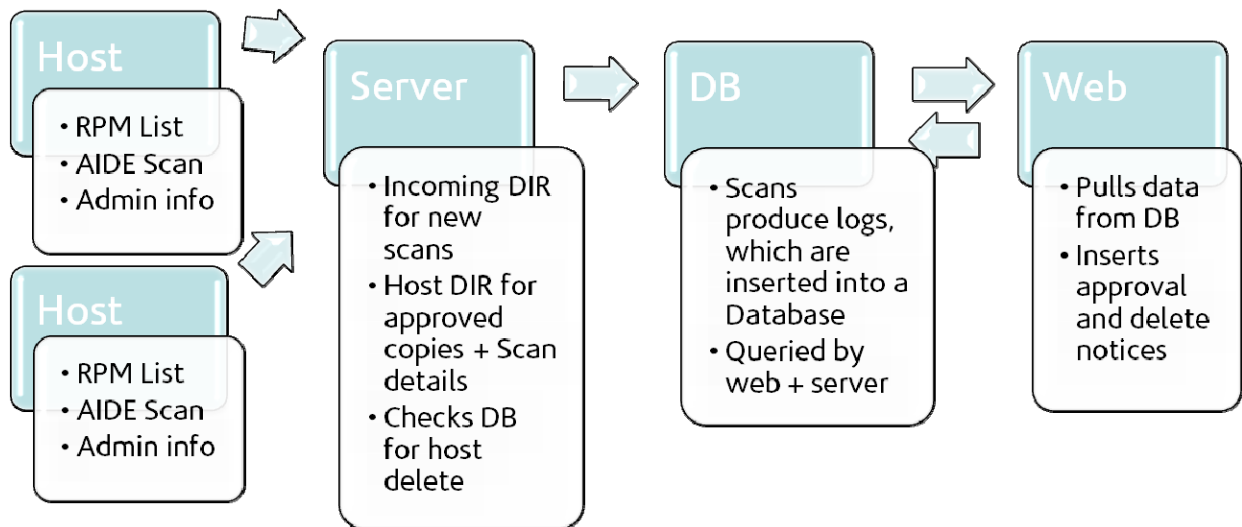


Figure 1.1

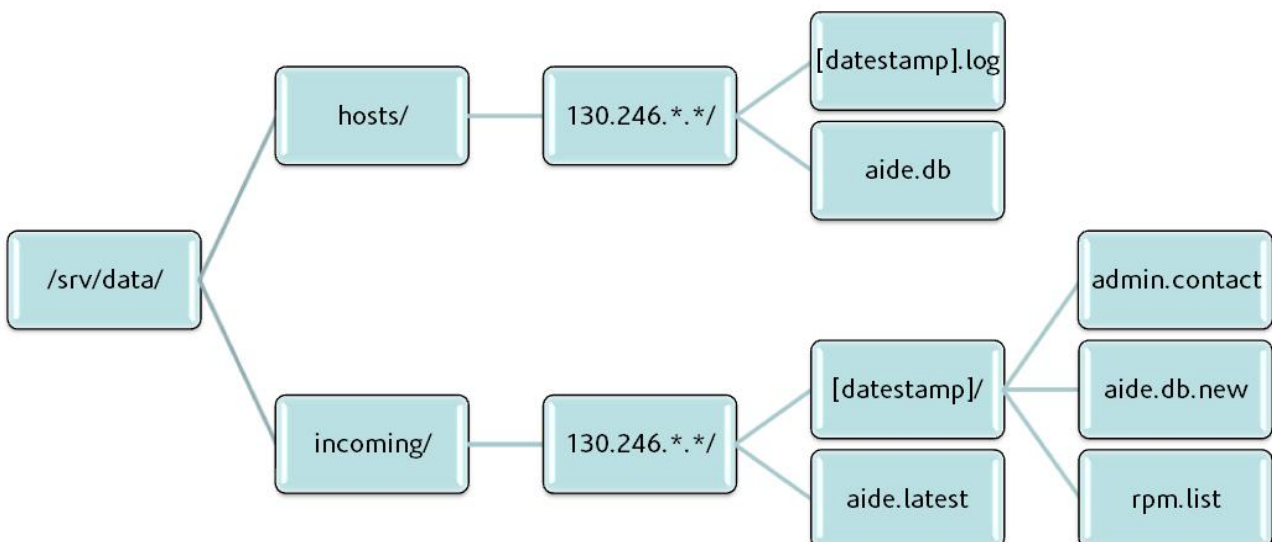


Figure 1.2

As Figure 1.1 shows, we have clients submitting multiple files to a server which then performs file comparisons at scheduled intervals, before storing the results in a database that is queried by a frontend web service. Figure 1.2 shows the directory structure on the server, with daily scans being stored in the 'incoming' directory and the known approved state of the system being stored under the 'hosts' directory, along with scan results. Directories are generated by the server side 'SCP wrapper script' and maintenance is performed by a monthly script to prevent the server disk from getting too full.

By providing a list of recently installed RPMs, AIDES can eliminate false positives generated when new packages are installed on a server. The client provides a list of the files which have changed as a result of recent applications being updated, as well as those which have been subsequently modified. The last part is important because simply ruling out modified files based on RPM installations gives an attacker the opportunity to modify files it knows will have recently changed. The RPM application has a verification flag which AIDES uses to determine, at scan time, whether recently installed or modified files are conformant with the package that installed them, and submits the data to the server.

The final requirement for the service was to allow system administrators to submit multiple system snapshots in a 24 hour period. This was implemented so that approvals could be made immediately when a series of packages were updated and the process of using that as a baseline could commence within the next hour instead of waiting until the next daily scan.

Deploying AIDES

With the enhancements necessary to deliver a product that can adequately meet the requirements of SCT and the e-Science Security policies, an RPM deployment mechanism was used. RPM (Red Hat Package Manager) [2] is a container which holds a series of script and a package payload to be installed on a host. AIDES consists of three RPMs; hpc-aide-client, hpc-aide-server, and hpc-aide-conf. The aide.conf file was separated from the main RPM in an effort to allow more flexibility in rapidly deploying configuration updates without altering the scripts used to drive the service. The hpc-aide-client package contains the necessary files to support the daily scan of a system and a client component of the secure copy mechanism of delivering the data to the AIDES server. hpc-aide-server contains all the logic and processing to drive the backend of the service.

The AIDES Frontend

After ensuring that relevant and correct information is being collected from clients, the next step was to construct a frontend to the service where data could be reported to system administrators.

Advanced Intrusion Detection Environment Service

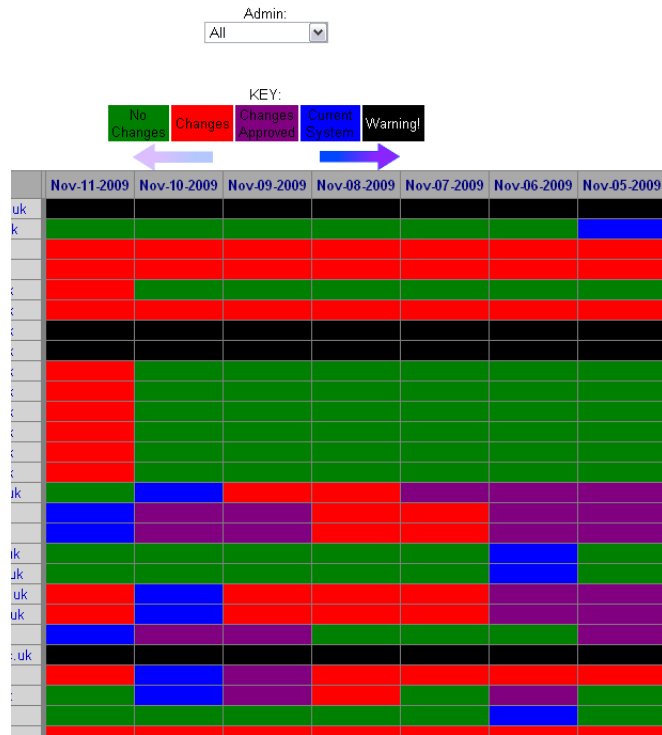


Figure 1.3

As figure 1.3 shows, there are several different states a host can possess on any given day. The key shows the most important distinctions between colours; green signifies that no changes occurred in the system from between the approved scan and the current submitted snapshot, red indicates that changes have been detected, and purple that the admin has approved the scan. The latest approval that is made by the admin will become the 'blue' snapshot of the system which will become the baseline for subsequent scans. If a daily scan entry is black then either a scan was not performed because the host was down, or because there was a serious concern with the data submitted to the server.

Access to the AIDES frontend is secured via signed digital certificate, with users only being able to view systems they are listed as an administrator for. There is also a super admin role which can see all scans. The system also provides historic data, stretching back up to 6 months worth of daily scans. An email alerting mechanism exists when a system has shown changes and these have not been approved for over three days.

Limitations and Future Development

Currently AIDES will generate false positives for kernel updates, as the process of installing a new kernel dynamically generates new (and very important) files which are not in its RPM manifest list. Additionally, by having an all encompassing single configuration file, as new services are rolled out, and more bespoke systems created, it becomes more of an effort to maintain the configuration.

The frontend webpage can only accommodate the latest scan which was performed on any given day, ideally the viewer should be able to drill down and view all scans which were performed. The coding is also somewhat antiquated, and where significant numbers of changes have been made on a system, the results page can take a long time to generate, even though it utilises asynchronous data retrieval. This is mainly due to the browser trying to render a very large html table.

The next revision of AIDES will see optimisations in the web frontend to ensure a better look and feel environment and more powerful asynchronous data transfer. Administrative tasks will also be added, augmenting the command line interface currently employed for maintenance and deletion. Lastly, a more robust mechanism needs to be implemented for client/server communication.

Effectiveness

As a service AIDES has proven a useful tool when seeing what kinds of files are changing on a system, giving SCT an ever clearer picture of how systems change over time. By recording the information centrally, AIDES acts as a reference point for a system if we believe it may have been compromised, allowing the administrator to choose from 6 month's worth of scans and determine when an attack may have commenced. As a tool, it is not designed to show what has changed in any given file, just the manner in which it changed. It is currently deployed on a small number of hosts to ensure the aide.conf file is as encompassing as possible, before being rolled out to critical servers.

3. Pakiti

Pakiti is a service which originally began as a development project at the STFC, being a successor to 'Yumit'. These services provide administrators with a list of package updates available for hosts they are responsible for, and highlight ones identified by Red Hat as critical security updates. Pakiti evolved Yumit's basic design by incorporating better filtering techniques and a more consistent overall design. Pakiti operates as a series of PHP scripts with a web frontend for reporting and utilises CURL for passing XML formatted data from client to host. Pakiti will also take input of operating system vulnerability status via XML generated by PHP and Perl scripts, which run daily. Everything is consolidated in a mysql database backend. Clients run a local script that uses data provided by yum or up2date to determine patch status of a client, and converts it to an XML file and submits it via CURL to the Pakiti Server.

Pakiti is now a Sourceforge project [3] being maintained by Michal Prochazka and Romain Wartel and currently available as version 2.1.2. This beta made significant changes to the way that Pakiti collects its host and vulnerability data, relying on the standardised Open Vulnerability and Assessment Language (OVAL) [4] data and using an unprivileged client to query currently installed packages instead of relying on applications like 'yum' and 'up2date'. This change was made to align Pakiti more as a tool for grid services which run very few root privilege applications, which meant that it no longer provided a list of available updates on a system and became much less useful as a tool for SCT.

Pakiti Reporting

Pakiti operates as a web frontend; displaying daily scans of package information submitted by hosts to a central sever. The website relays this information through a number of different views. The display of information is filtered by the assigned system administrator and subcategorized by Operating System version. Hostnames are coloured red if they have existing security patches due to be applied, or if they require a reboot because they are not running the latest kernel. Information can also be viewed by package, where all hosts which require a specified package can be displayed.

Maintaining Legacy Pakiti 1.0.1

Given the changes made in Pakiti 2, it was decided that SCT would maintain their own codebase of the older 1.0.1 version of Pakiti last modified in April 2006, implementing fixes and enhancing features while maintaining the core functionality needed to meet SCT's requirements. Even though the application itself is no more than a few files, in order to track changes and manage the project, the code was stored in SCT's Subversion service. From here we could commit or revert changes made to ensure that the production service was not impacted due to testing.

One of the first changes made to Pakiti was to more tightly integrate it into our server commissioning process, as it is an application that every machine needs to have installed. We wrote an RPM to contain the client and configure it based on key information contained on the system it was being installed on, such as the listed Administrator, and Operating System. By doing this, every new system added will be grouped by Administrator then sub-grouped by OS version. Originally the mechanism in the RPM would check an internal list to see which 'owner' Pakiti would associate the system with, but this became unmanageable as SCT grew. The solution was to simplify the process and capture information specified in /etc/MOTD which is generated on all SCT systems, that specifies the Administrator contact information. Additional efforts were also invested to add compatibility with Scientific Linux. Originally Pakiti was designed for the CERN version of Scientific Linux and made some assumptions about where vulnerability information would come from. This data was not very useful, so we developed a script that would generate the same information locally. RHEL5 was also not supported with the original application, so this had to be coded in.

In order to protect the webpage, 'GridSite' [5] security was implemented, restricting access to a white list of user certificates. Finally, a feature inexplicably missing from the original application, the ability

to delete host entries which have stopped reporting, was added. Ongoing maintenance is still necessary with Pakiti, to ensure that it recognises the latest releases of operating systems. Currently 542 hosts are registered with the SCT Pakiti service.

Effectiveness

By reporting information in this fashion the SCT System Administrator, who may be responsible for a large number of systems, can very easily view security patch data and act accordingly. Having a central store allows rapid response and oversight when addressing vulnerabilities. This service is ingrained in the core suite of applications required to deploy a host in the SCT infrastructure.

4. Syslog-NG

As part of SCT's standard Operating System install procedure the 'syslog' application [6], provided by Red Hat, is configured on each machine to log locally. Syslog is used by all applications that generate logs on a Linux based system, collating them into application types and channels and typically storing them in /var/log. The problem with this is that if a host fails or becomes compromised, the forensics prove difficult when we cannot access or trust the logs contained on the server. One solution to this is to rely on an application called Syslog-NG [7], which aggregates logs onto a central server. This comes in two variants, one being a free open source product and the other a Premium Edition (PE) that provides extra functionality. Syslog-NG can also act as a replacement client to Syslog, however for our purposes, and to reduce administrative overhead, we kept the original Syslog running on each client.

While aggregating logs can be an effective strategy to provide remote forensics and monitoring for potential problems on systems, the result is a vast hierarchy of directories separated by date and hostname that can prove very difficult to search through and find results when over 500 machines are being monitored. In light of this a clear requirement to improve the way searches could be performed and improving data accessibility was set. We also saw the risk of a single point of failure and, since our network monitoring showed no bottlenecks, we set a requirement to provide redundancy. Coupling this with the previous requirement signified that a new mechanism for recording logs would need to be developed.

Syslog-NG has the capability of acting both as a client and a server, so the mechanism for passing along logs from a Syslog-NG server to some kind of data repository did not prove a difficult challenge. The formatting of the data and the mechanism for passing it along was a little more difficult.

Development Proposal

In order to mitigate the single point of failure currently existing in the Syslog-NG infrastructure, it was decided to install a second Syslog-NG server. Each SCT host would report their logs to both servers giving not only the capability of fault tolerance but also a mechanism to allow maintenance to be performed on one Syslog-NG server at a time.

One important caveat of this system was that there would be no guarantee that each Syslog-NG server will be a mirror of the other. Allowing for maintenance and downtime will result in the servers containing different amounts of logs. To address this issue it was proposed that the logs should be inserted into a database so that there was a unified repository. In order to provide this, a mechanism had to be defined to allow duplicate log entries to be removed before being inserted into the database. There would also need to be additional fault tolerance steps given that the database could represent a single point of failure.

SCT Syslog-NG Mechanism

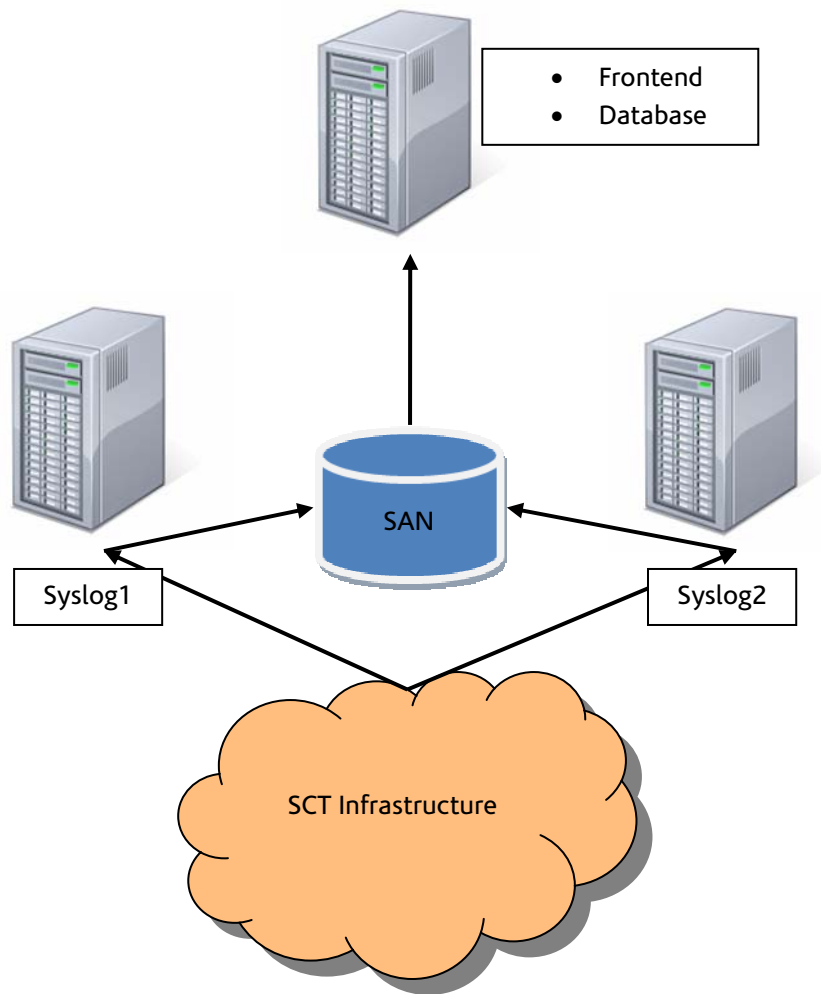


Figure 2.1

For every log that a host in the SCT Infrastructure generates, it is forwarded to both Syslog1 and Syslog2. These two Syslog-NG servers keep a local copy of the logs in a hierarchical directory structure based on the day the log was sent, and also forward a minutely sample of all logs received to a shared storage repository on the SCT Storage Area Network. The motivation for this was to ensure that logs can still be queued up in the event that the database server is down or having maintenance performed. While the SAN still provides a point of failure, it is not expected to experience as much service disruption as any one other component in the service. Additionally, even in the event of a SAN failure, the logs are queued up on the Syslog-NG servers until it becomes available again.

After 10 minutes worth of logs have been collated, a script runs on the frontend server, to pull in the logs, remove duplicate entries, and then insert them into the Syslog database. As the volume of traffic involved in capturing logs can be very high, the database is divided into weekly tables, with old tables being compressed to conserve storage.

Syslog-NG Frontend

To provide a frontend to the SCT Syslog-NG service, giving a more convenient interface than using `grep` to search through logs on the individual Syslog-NG servers, we utilise an application called PHP-Syslog-

NG [8]. This designs the database schema which is compatible with Syslog-NG and provides a frontend search engine for logs, with graphing and 'latest entries' features.

The screenshot displays the Syslog-NG search interface. At the top, there is a navigation bar with 'Logout', 'Search', 'Config', 'Help', and 'About'. Below this, a dropdown menu shows 'SELECT TABLE: logs'. The main area is titled 'USING CACHE TO POPULATE HOST, FACILITY, AND PROGRAM FIELDS. Cache last updated on 2009-12-10 10:00:01.' It is divided into several sections: 'HOSTS: 450' with 'Include' and 'Exclude' radio buttons, 'PROGRAMS: 224' with similar radio buttons, 'SYSLOG FACILITY:' with a list of facilities like 'auth', 'authpriv', 'cron', 'daemon', 'ftp', 'kern', 'local0', 'local1', and 'SYSLOG PRIORITY:' with a list of priorities like 'debug', 'info', 'notice', 'warning', 'err', 'crit', 'alert', 'emerg'. There are also 'RegExp Matching?' checkboxes and 'Hostname match'/'Program match' text boxes. A date and time range selector is present with 'From:' and 'To:' fields. On the right, there are settings for 'RECORDS PER PAGE' (50), 'TopX' (10), 'ORDER BY' (datetime), and 'SEARCH ORDER' (DESC). Below these are 'SEARCH MESSAGE:' fields with 'Exclude' and 'RegExp' checkboxes, and an 'AND' operator. At the bottom, there is a 'COLLAPSE IDENTICAL MESSAGES INTO ONE LINE:' checkbox and buttons for 'Search', 'Tail', 'Graph', and 'Reset'. A 'Common Graphs:' dropdown is also visible. The footer indicates 'Executed in 0.037094116210930 seconds'.

Figure 2.2

As Figure 2.2 shows, the frontend interface gives comprehensive filtering for search queries, tailored to the format logs are delivered in. Searches can be delivered in much shorter timeframes when constraining the search, and significantly faster than using grep on the Syslog-NG servers. In order to comply with our SCT policies on service access, we use GridSite security to protect access to the frontend based on User Certificate.

Effectiveness

PHP-Syslog-NG has demonstrated an invaluable tool when investigating security incidents, providing a powerful search tool. While searches can be slow when scanning over a large period of time, this is because the Mysql database backend contains 110GB of compressed log data spanning 868,878,122 records at the time of writing.

5. Conclusions

The three open source applications outlined in this paper, which have been transformed into robust services that meet the requirements of SCT, have proven themselves to be valuable assists in the System Administrator's arsenal. They cater for a large number of hosts and can display very accurate and trusted information about a diverse infrastructure. Discretely, each service can provide very useful information, but combined they greatly facilitate the forensic examination of an infrastructure which may have been compromised. As SCT's infrastructure grows and diversifies to meet the demands of grid computing and bespoke services, these tools will scale alongside, ensuring the environment remains monitored and understood.

6. References

- [1] Advanced Intrusion Detection Environment website:
<http://www.cs.tut.fi/~rammer/aide.html>
- [2] Red Hat Package Manager website:
<http://www.rpm.org/>
- [3] Pakiti Sourceforge page:
<http://pakiti.sourceforge.net/>
- [4] Open Vulnerability and Assessment Language (OVAL):
<http://oval.mitre.org/>
- [5] Grid Security for the Web
<http://www.gridsite.org/>
- [6] Syslog
<http://en.wikipedia.org/wiki/Syslog>
- [7] Syslog-NG website
<http://www.balabit.com/network-security/syslog-ng/>
- [8] PHP-Syslog-NG
<http://code.google.com/p/php-syslog-ng/>