SANS Institute
InfoSec Reading Room

Creating a Bastioned Centralized Audit Server with GroundWork Open Source Log Monitoring for Event Signatures

Organizations spend millions of dollars each year to implement Auditing services and personnel who will monitor the enterprise deployment. Unfortunately, the technical solutions usually lag behind in meeting all regulations laid forth. The designs almost never meet all of the expectations and cost projections. The resolution prescribed here is independent of a network size, site number or management concept. Whether the organization uses centralized, decentralized or hybrid models this design will work. The tools provi...

Creating a Bastioned Centralized Audit Server with GroundWork Open Source Log Monitoring for Event Signatures

GIAC (GSEC) Gold Certification

Author: Christopher S. Duffy. Christopher.S.Duffy@gmail.com
Advisor: Antonios Atlasis

Accepted: January 30th, 2013

Abstract

Organizations spend millions of dollars each year to implement Auditing services and personnel who will monitor the enterprise deployment. Unfortunately, the technical solutions usually lag behind in meeting all regulations laid forth. The designs almost never meet all of the expectations and cost projections. The resolution prescribed here is independent of a network size, site number or management concept. Whether the organization uses centralized, decentralized or hybrid models this design will work. The tools provided are a baseline for Auditing and event signature monitoring. This solution can be tailored to fit any organization.

1. Introduction

Setting up an Audit server is more than just pulling a piece of hardware off a shelf, slapping it in a rack, hooking it up to the network and off to work it goes. A number of considerations have to be made about how it is going to operate. Who the Audit server is going to serve? What else is the Audit server going to do? Where is the hardware going to be placed? Who is going to have access to the server? How long are the logs retained? Will the logs be reviewed, if they are by whom? Finally, why is the organization deploying it and what regulation(s) is it trying to satisfy? These questions must be answered, else the deployment of the server is not answering the call it was intended for.

The centralized Audit server is a piece of the security puzzle. This piece will be used to ensure business applications are functioning properly. The logs presented will show which systems are beginning to fail or already have failed. The logs will provide historical data and a baseline of normal organizational traffic that traverses the systems. The Audit server will allow the organization to respond to events in real time, when they happen, not if they happen. Finally, the Audit server will prepare the organization to be not only forensically ready, but legally ready to prosecute after a malicious event.

The Audit server gives the organization a means to address all three security objectives if implemented properly. These objectives are the Confidentiality, Integrity and Availability (CIA) of information which are the key stone of security design (Tipton, 2010, p. 4). The confidentiality of the data is the guarding of data so that it is not known to unauthorized users. The integrity of the information is to prevent the unauthorized modification of that data. Lastly, the availability of the items ensures that when someone needs access to that information, they can get to it.

The confidentiality of logging data is maintained by the use of restricted access, encrypted storage, protected transmission and mutually authenticating the traffic to and from the Audit server (Tipton, 2010, p. 322). The integrity of information is enabled by offloading logs from end devices to a centralized storage location and digitally signing them. This notifies reviewers of a modification of that original data if the asset is compromised. The last objective availability is achieved by secured retention of the log data. This retention period is determined by the organization, an example of this would be at least one month online and five years offline.

None of this logging data is any good to an organization if no one ever looks at it. The problem is how does a security analysts review the enormous amount of data produced just by a single system? In just one minute the average application server will produce at least 30 log entries. Many of these logs can contain benign information, others malicious. When the number of servers is multiplied by the hundreds or even thousands, it is no wonder why personnel cannot keep up.

Using multiple monitors to review multiple system logs is not only problematic, it is not sustainable. According to Tipton (2010) "The Police Scientific Organization has done research on the number of monitors one control center operator can handle effectively. They indicated that no more than four to five monitors per person should be allocated. This is supported by their research on observing observers viewing one, four, six, and nine monitors which showed accuracy detection scores of 85%, 74%, 58%, and 53%, respectively. Clearly detection rates go down, as people have to cover more information" (p.620). To combat this problem, log view reduction tools need to be used to alert on events of interest.

2. The Solution

This example will show how to enable signature based log review with the GroundWork Open Source (GWOS) monitoring solution (GroundWork, Inc, 2012) on a centralized Audit server. This solution incorporates a number of tools together to visualize events as they happen in near real time. The version presented here is GWOS 5.3 which is comprised of Personal Home Page or Hypertext Preprocessor language (PHP), Apache, MySQL, Nagios and NagVis. The newer version of GWOS runs JBoss Application Server, Apache, Nagios, Cacti and MySQL. With custom definitions, the system can review logs for specific signatures and alert on them.

This signature based detection is great for notification of malicious events as they happen, but it can lead to false positives. The use of GWOS in this manner should be paired with an additional log monitoring solution that alerts on anomalous traffic. The pairing of these tools would allow an organization to quickly identify events of interest.

To increase security of the Audit server, access will be restricted. If an event is questionable, the Information Assurance (IA) team and or security analysts may have to look at

the actual logs. They can review the logs for further detail manually. This requires specialized and restricted access to the information through 'sudo'.

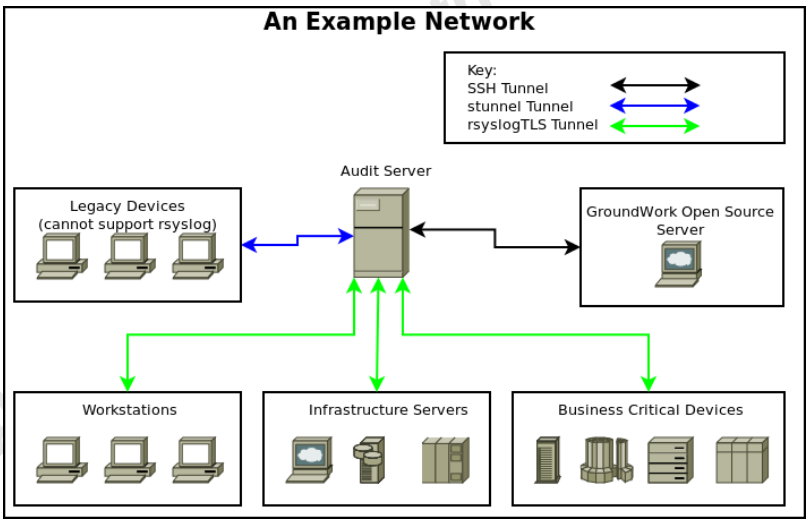
The Audit server is a prime target for malicious events. Each service, program, or package installed increases the number of vulnerabilities that it contains. To combat this problem, the Audit server will be bastioned to minimize its vulnerability footprint. For those services and programs that cannot be removed, additional security tools will be used. The server will have Internet Protocol Tables (IPTables) software firewall turned on and tuned. IPTables will help prevent many forms of stealth port mapping through network mapper (nmap) and unauthorized service access. In addition to IPTables, Security Enhanced Linux (SELinux) a Mandatory Access Control (MAC) capable operating system (Harris, 2010, p. 212) which can be enabled to curtail certain types of buffer overflow attacks and service breakouts.

This protection is created by stopping an associated account of the broken service or subject from enacting objects it is not entitled to. This problem is shown by Harper, Harris, Ness, Eagle, Lenkey & Williams (2010) "When dealing with buffer overflows, there are basically three things that can happen. The first is a denial of service. As we saw previously, it is really easy to get a segmentation fault when dealing with process memory. However, it's possible that is the best thing that can happen to a software developer in this situation, because a crashed program will draw attention. The other alternatives are silent and much worse. The second thing that can happen when a buffer overflow occurs is that the eip can be controlled to execute malicious code at the user level of access. This happens when the vulnerable program is running at the user level of privilege. The third and absolutely worst thing that can happen when a buffer overflow occurs is that the EIP can be controlled to execute malicious code at the system or root level." (p. 208). SELinux does not help protect against Denial of Service (DoS) created by a buffer overflow attack, but it does protect against the other two forms.

The last items to mention about securing the Audit server is service deactivation, physical access restriction and Anti-Virus (AV) software. The AV software and service monitoring will help protect against the buffer overflow based DoS. The physical restriction and service deactivation provide the final layer of protection on the Audit server ensuring a solid Defense in Depth strategy at the system level (Harris, 2010, p. 39).

Network Time Protocol (NTP) is vital to a logging server, as it provides an official time of events. The system clocks of hosts that forward logs to the Audit server must match the Audit server. This is to prevent a skew in time between devices during an event. If the records were called into question during a trial, they may be disregarded if there are inconsistencies between devices because of configuration failures. A malicious user may attempt to alter logs on a device that sends its logs to the Audit server. Tracking of the time of an event and narrowing the changes made to end device logs will allow true event logs to be filtered. The logs on the Audit server will delineate the actual events that occurred and what tracks were covered.

All logs sent to the Audit server must be mutually authenticated and encrypted through tools like stunnel Secure Socket Layer (SSL)/Transport Layer Security (TLS) or rsyslog TLS. The figure below provides an example network that depicts these communication forms.



A network configuration like this mitigates log partition overflow with spoofed syslog messages, it also mitigates log sniffing or Man-in-The Middle (MiTM) attacks. After all these items are configured a full scan of the system using Nessus (Nessus Vulnerability Scanner, 2012) or Nexpose (Vulnerability Management Software - Nexpose, 2012) scanners will identify any shortcomings. Use of these scanners will not help in the removal of all vulnerabilities, but it does point to known vulnerabilities that are within the scanner's databases (Kennedy, O'Gorman, Kearns & Aharoni, 2011, p. 36). Because of this, it is encouraged to use multiple vulnerability scanners to ensure a thorough review is completed. Configuration of these tools and more can be found throughout the web and as such will not be covered here.

This example make no assumptions about the reader's Linux or GWOS knowledge or what resources they have available to them. In addition, each organization is different, and its needs and regulations will drive changes to the baseline provided here. Organizationally, specific details will have to be done by the developer and engineers at that location. This document provides many solutions, just not all the solutions. Suggested reading material for further securing both the GWOS and the Audit server are listed in Appendix B.

3. Log Organizing Scripts Operational Theory

The processing of logs on the Audit server has multiple steps. Each of the follow-on processes is taken care of by shell scripts. These scripts were designed to be simple to understand and read so that changes can be made as necessary for an organization. This example uses Bash shell because it requires no other features and would allow an organization to further bastion a system. Though Perl is used for the 'check_log2.pl' script for the GWOS signature test, there are no particular special modules within that script. Some of these Bash scripts may require such modules if it was completed in Perl instead of Bash. If the administrator chooses, a rewrite of all the scripts in a different language is possible. Bash was chosen to reduce the possible vulnerability footprint and to provide an easy human readable example for this configuration.

3.1. Log Control Stages

The first thing that happens is the Audit server's rsyslog daemon receives logs from hosts around the network as seen in the figure below. These logs can be encrypted through an SSL or TLS tunnel or they could come in unencrypted depending on how the organization chose to proceed. The logs are either decrypted and passed to '/var/log/messages' or sent in their unencrypted form. All logs have to be sent to '/var/log/messages' so they can be captured by the GWOS signatures tests and follow-on scripts.

At the end of every hour the logs are moved to a temporary file and parsed by the 'log_move' script. All of the following scripts presented here can be found in Appendix A. The system dictionary is used to break the logs out to system specific log files in '/var/log/systems/<system_name>'. These files are the ones that GWOS will be able to parse for signatures of events and flag as required.

The ‘/var/log/systems/<system_name>‘ files will be moved at the end of every day to ‘/var/log/archive/<system_name>‘. This is to prevent GWOS from reviewing extremely large files, taking up Central Processing Unit (CPU) clock cycles on the Audit server. Once the files are reviewed at the interval set by the ‘Notification Check Interval’, GWOS will no longer flag those events, but it will still have to parse through the full file to get to the byte location bookmark. It is important that the ‘Notification Check Interval’ on the Audit server's ‘Service Profile’ page ‘Service Detail’ tab be more frequent than the ‘log_archive’ script. When configuring the Audit server ‘Host Profile’ later, keep this in mind.

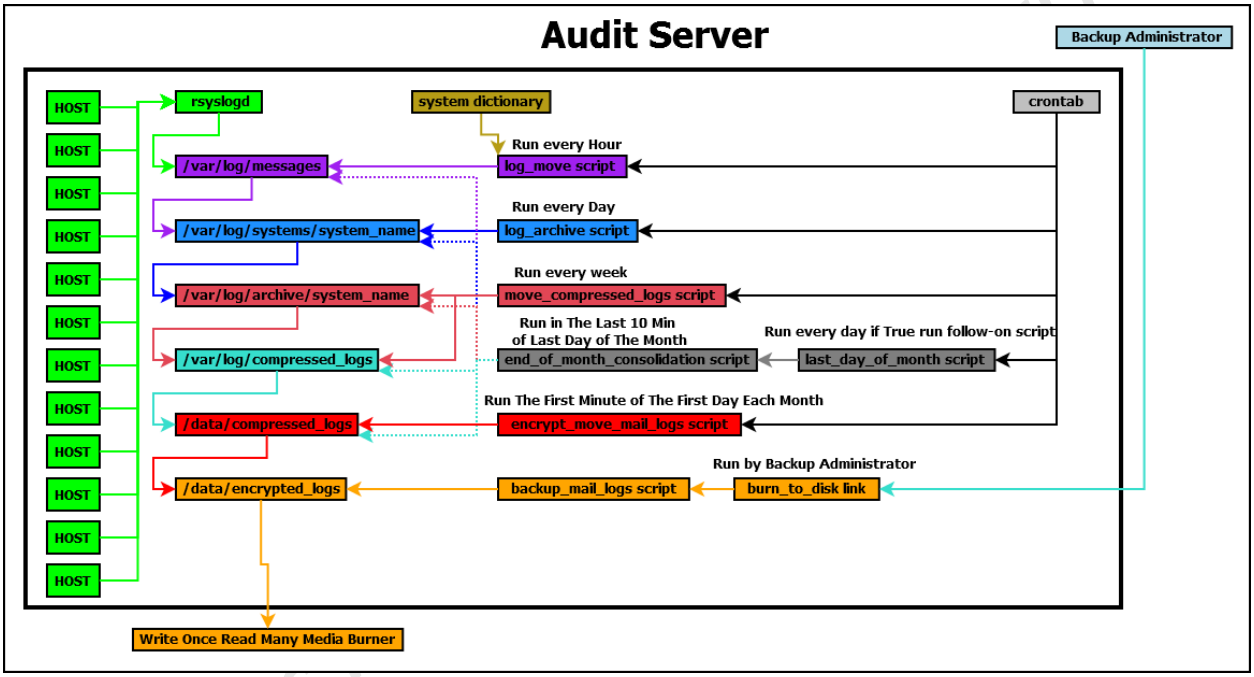
At the end of every week, the ‘move_compressed_logs’ script moves the ‘/var/log/archive/<system_name>‘ files to the ‘/var/log/compressed_logs’ directory. The logs are compressed there and moved to the ‘/data/compressed_logs’ directory. The logs remain there till encrypted and digitally signed then moved to the ‘/data/encrypted_logs’ at the first minute on the first day of the new month. The logs remain here till they are burned to disk by the designated technician.

The ‘end_of_month_consolidation’ script runs at the end of every month and completes the cycle of all other scripts again. This review ensures that the logs that had not been moved prior to the end of month are moved. Some logs will still be missed because the timing between the ‘cron’ and the scripts processing time cannot be absolute. The trick is to test and attempt to get it as close as possible. Keep in mind that if an incident happens at the end of the month, the next month’s logs may have to be reviewed for the data. The amount of skew is minimal, but it is present.

The ‘encrypt_move_mail_logs’ script takes all the compressed logs in the ‘/data/compressed_logs’ directory and combines them up into a Tape ARchive (tar). Once the logs are tarred up they are encrypted and digitally signed by the Audit server and placed into ‘/data/encrypted_logs’ directory. The script also sends out a notification email to the administration team and the IA office that it is time for the logs to be burned to disk.

The backup administrators will have to check out a media burner and the physical access keys to the rack with the Audit server in it. Once the backup administrator logs into the Audit server, he or she runs the ‘burn_to_disk’ link. This executes the ‘backup_mail_logs’ script and checks for the data in the ‘/data/encrypted_logs’. Once the data is found, the logs are checked to

see if they are encrypted and then burned to a disk. The disk is then cataloged, filed and stored in a safe for the regulatory period of time. The next sections explain in depth how each of the scripts work and the actually processing within them and some associated service configurations. The actual scripts can be found in Appendix A of this document for both concise delivery of information and ease of reference.



3.1.1. Create Specific log directories

These specific log directories are going to be used for moving and storing the log files. All of these files and directories will only allow root to access them with the exception of one. The '/var/log/systems' directory must allow a user account to view the contents of the file that is in it. These files can only be accessed if a user knows the exact file name(s) within the directory. Since GWOS must have access to the files within this directory, the optimal permission structure is setting the execute bit for groups, other users and read write and execute for the owner of the directory and files. The log files will be generated and updated when the log organizing scripts are run. All other locations only require root and 'sudo' authorized users' read, write and execute access. Root will own all of the directories and files that are created. With that information create the '/var/log/systems' directory with '711' privileges and create '/var/log/archive', '/var/log/compressed_logs', '/data/compressed_logs' and '/data/encrypted_logs' directories with '700' level privileges.

3.1.2. Configuring the Mail Transport Agent

The Mail Transport Agent (MTA) is used by the Audit server to send an email to a target address. This is a recurring email that will be done every month to alert the data custodian or backup administrator and IA office to offload previous month's logs. Once the logs have been offloaded, the server will then email all respective parties again that the logs have been either been offloaded or failed to burn to a disk.

This feature is dependent on the organization's security policy and whether they want it enabled. The benefit of using the service is that parties have an audit trail of log offload practices or proof that they are being done or not. This could be beneficial for organization audits and accreditation requirements. The side effect of this configuration is that it does open up another service vulnerability to the Audit server. The organization itself will have to determine if this risk is acceptable and that the benefits outweigh the costs.

The Audit server will only be acting as a client, using sendmail to push the email to the target mail server. The Audit server will have the capability of the sendmail daemon service disabled within '/etc/sysconfig/sendmail' file. After disabling the daemon capability, the target email server will be listed in '/etc/mail/submit.cf' file.

Edit the '/etc/sysconfig/sendmail' configuration file:

```
vim /etc/sysconfig/sendmail
```

Turn of the Daemon capability on the server and write the file:

```
DAEMON=no
```

Edit the '/etc/mail/submit.cf' submissions configuration file:

```
vim /etc/mail/submit
```

Set the value of 'D{MTAHost}' to the mail server and write the file:

```
D{MTAHost}mail.example.com
```

The scripts that deal with log encryption and log movement will send an email out to the appropriate addresses. This ensures that the Audit server notifies the personnel when a task is complete, ready to be completed or failed. This helps prevent a backup administrator from going

to offload logs that are not ready for burning. Though would be a minor issue, it could start a round of unnecessary troubleshooting.

3.1.3. Create System Specific System Listing and Log Location File

When choosing the '/var/log/systems/<system_name>' files, it can be beneficial to group similar business systems to the same log file. This reduces the number of system specific signatures that would have to be created. This design also enables GWOS to process events faster.

The GWOS server would still flag the event to the particular machine, so the data residing in the same location should have little impact on tracing signatures. At the same time, keep systems segregated by either their purpose or business operational attachment. Otherwise, when an auditor has to manually parse the logs, he or she may have to look in unrelated locations. There will be times when this is not possible and separate log files will be required. Additional information will be provided in section 5 about feeding multiple sites with Audit data. This will affect the way the directories and files are organized on all servers in the end. A proper enterprise survey must be completed prior to implementation. This will determine the path that best meets the needs and goals of the organization.

3.1.4. Create a Log Parsing Temporary File Location

This directory in '/root/systems' will hold the 'temp_messages' file that is a direct move from '/var/log/messages'. This directory's only purpose is to deal with temporary files in a segregated location. This file will have to be reviewed when initially configuring the Audit system deployment. Signatures of system breakout will be missed by administrators and reviewing this temp log will allow those missed signatures to be captured in future passes.

This means that as hosts are added to the Audit server. The temp file must be checked to make sure that logs are not being missed in transfer. These problems will reduce over time as the complete Audit system becomes more stable.

3.1.5. Create The System Specific Log breakout script

The first script to be created will break apart the '/var/log/messages' file and pull all the relevant data out per system. This is called the '/root/log_move' script which looks at the dictionary listing of system names and final log location in '/root/system' file. This file will

contain a system name and final log location separated by a colon such as this 'hostname:/var/log/systems/hostname'. There can be two entries per system, a hostname entry and an IP address entry.

The IP address entry is optional but it would be best to capture systems that have static IPs assigned to them, otherwise use the DNS registered Fully Qualified Domain Name (FQDN). The call for this is sometimes poorly coded third party tools will alert on events via IP address and not the system name. It will not be possible to configure signatures based on dynamic IP addresses because of their ever changing nature. Critical systems typically do not have dynamic IPs so this is usually not a problem. For the sake of having a complete event picture static IPs must be added as well.

If the log breakout script is started manually and not by 'cron', it will provide a metric report as it runs to show how far it has progressed. This script, along with the others, will be placed in the root's 'crontab' to ensure that it runs at the appropriate interval. As mentioned before, to not impede performance of the rsyslog daemon, the '/var/log/messages' is copied over to a temporary location in '/root/systems/temp_messages', where the file is parsed. The rsyslog daemon is then restarted to ensure that the '/var/log/messages' log file is recreated. This step also prevents logs from being missed by constrained input/output processing of '/var/log/messages'.

A temporary file is used to transition the logs through a 'sed' routine, but it is deleted when the script is finished. The '/root/systems/temp_messages' file remains so that security administrators can review it and ensure all logs are being picked up. Any remaining logs just need their signature added to the '/root/system' file. This is especially important to review when a new system is added or when the Audit server is first brought online.

The location that this script sets for the files is where GWOS will parse the logs for signatures. These files will be appended to every hour and GWOS will access them in the time limit set by the 'Notification Check Interval' under the Audit server service check profile 'Service Details'. After remaining in this location for one day, the logs will be moved to the archived position.

3.1.6. Create The System Archive Script

The Audit logs will remain in the system specific log folder for one day. After that one day period, it is assumed all signatures that were necessary have been captured. The logs are moved to an archive location on the server, which is just a different directory. This action is done by the 'log_archive' script. The logs remain un-tarred and un-encrypted until half an hour before the end of the week. This is done to allow any signatures found to be researched prior to being tarred up.

The last step this script does is null, or truncate, the reference file GWOS uses to keep track of its location searching through the logs. This is done to prevent this file from growing unnecessarily large over a period of time and then locking up the system. In theory, it would take a very long time to accomplish this, but it is something that could easily be overlooked when a crash does happen.

3.1.7. Create The Log Compression Script

This script's purpose is to take the logs and compress them every week. The logs are left on the system to allow the IA office or the security administrators continued access as needed. The intent is to leave the logs on the system for one month to ensure that any incidents can be quickly followed up, though this can be adjusted in the 'crontab' as needed. The directory that the logs will be moved to is first checked to see if any compressed logs are present. This is a safety check to make sure nothing was missed during the last move. If compressed logs are present, they are first moved to the '/data/compressed_logs' directory. This safety check ensures that no half-finished cycles due to crashes or performance hiccups cause files to be compressed multiple times.

Once this check is complete, the script takes logs out of the '/var/log/archive' directory and moves them to '/var/log/compressed_logs'. In this location, the logs are compressed and dated, then moved to the '/data/compressed_logs' directory. The logs remain here until they are ready to be encrypted and burned to a disk.

3.1.8. Create The Last Day of the Month Reference Script

All the processes that have happened prior to this one are done on of the end of every hour, the end of every day and the end of every week. The end of the last week of a month rarely is the actual end of the month. This means that data that should be written at the end of month to the encrypted log tar ball may get missed.

Unfortunately, ‘cron’ does not have a clean way of determining if “today” is the end of the month, especially during a leap year. This script does a simple Boolean check to determine if tomorrow’s day of the month is less than today’s. If it is, the script triggers a return of true which activates the ‘/root/end_of_month_consolidation’.

3.1.9. Create The End of Month Cleanup script

This script performs the functions of the ‘/root/log_move’, the ‘/root/log_archive’ and the ‘/root/move_compressed_logs’ scripts in succession within one package. The purpose of this script is to move all the yet to be processed logs for the month over to the ‘/data/compressed_logs’ directory. This is all accomplished in the last ten minutes of the current month. This seems to be plenty of time for full processing of over one thousand extremely chatty systems on archaic hardware. Less time can be given if the hardware and network is more robust or fewer systems are served. As always, testing to the organizations specifics will dictate the timing of this process. This does mean a few logs may end up being on the next month’s burn, but there has to be some lag time for full processing.

3.1.10. Create Emails for Log Offload and Offload Completion

The next two scripts, ‘/root/encrypt_move_mail_logs’ and ‘/root/backup_mail_logs’, will need a generated email to be pushed to its targets. The first email will inform the security administrators and the IA team that the logs are ready to be burnt to a disk. This is to ensure that the logs are being offloaded in a timely manner so that they are available as necessary during the mandatory retention time frame.

The second email is for when the logs are actually burnt to a disk and provided to the IA and system and security administrative offices. This ensures that there is no repudiation of the logs being burned to disk or not. It also ensures that IA is aware that the logs have been removed from the system and can be expecting to see them. Finally, this provides historical data for regulatory compliance.

The last email is for failed log burns for whatever reason. The email will be sent off to both the administrators and the IA team. This again assures there is no repudiation of log failure and all information can be documented. This type of event is usually correctable but only if personnel know it has happened.

3.1.11. Create Script to Tar and Encrypt Logs then alert Administrators

This script provides the capability to take all logs stored within the '/data/compressed_logs' directory and encrypt them. The first step done by this script is the emailing of the security administrators and IA team. This is done in case the script ever fails, then an administrator still knows that the logs are supposed to be present. The administrator can then assess why the logs failed to move and attend to the problem at hand. Once the logs are encrypted, digitally signed and dated, they are moved to the '/data/encrypted_logs' directory. This is where the logs are stored until they are burned to disk by the '/root/backup_mail_logs' script called by the '/data/burn_to_disk' link.

3.1.12. Create Audit Log Backup Script

The backup script is placed in the '/root' directory and symbolically linked to '/data/burn_to_disk'. This allows users who do not have full-fledged root access, the ability to run the backups to a disk. There has to be some level of trust with the administrator or data custodian who is offloading the logs to disk. This is because of the provided access to the critical resources of the organization. In addition, when the logs are written to disk, they are essentially the only copy of the logs. If the data custodian misplaces or destroys the disk, the logs are lost forever unless they are forensically recovered from the Audit server.

To help ensure there is accountability when the logs are moved off the system, physical access should be restricted until actual time of log offload. A solid key checkout policy and control of media burners is a must to prevent the compromise of such critical data. Standard critical resource control policies should be in place for the Audit server.

If the logs are present and encrypted, they are turned into an International Organization for Standardization (ISO) 9660 image that is readable by Windows, Linux and UNIX systems. The script automatically detects the USB media burner. The script then burns the ISO9660 image to the media and closes the burn. The server then validates the presences of the logs on the disk before destroying the resident copy on the server. The disk is then ejected and ready for filing and storage.

Once the burn script is created, its features will have to be accessible by the backup administrators. The '/etc/sudoers' file must be setup to allow the items that are within the script to work. The 'sudo' authorizations will be different for each organization so some testing will be

required. Plenty of tutorials are available online that show how to configure 'sudo' rights for users and groups as such this information is omitted from this example.

> As a note, the '/root/backup_mail_logs' script utilizes 'cdrecord' which is an older program for disk burning. Testing must be completed to verify that disks can be burned with this utility and with the USB based drive owned by the organization. This has to be completed before putting the Audit server online. Otherwise, if this does not work the script will clean-up the files at the end of the job, and there will be no recorded events. The script attempts to check if the material burned effectively but it is still hardware dependent.

3.1.13. Configure crontab for Audit Server Log Control

Now that the scripts have been created for the Audit server, they have to be setup to run in a timely manner. Each script is designed to work in a specific order. If that order is broken, then entire process will break down. In addition, each script must have enough time to finish its task, else the results of the Audit log retention will be skewed. The configuration of the root user's 'crontab' in Appendix A provides the results desired, below is the specifics of how the tab is setup.

The '/root/log_move' script runs at the end of every hour. The '/root/log_archiver' script runs at the end of every day. The '/root/move_compressed_logs' script runs at the end of every week. The '/root/last_day_of_month' script runs at the end of every day and if it returns a positive result, the '/root/end_of_month_consolidation' script runs. Lastly, the '/root/encrypt_move_mail_logs' script runs on the first minute of the first day of the month. With the dictionary file, scripts and 'crontab' configured, any logs currently going to the Audit server will be processed appropriately.

3.1.14. Finalizing the Audit Server

At the end of the development cycle a final review must be completed to ensure there are no loose ends. With the Audit server working correctly, it is time to complete the hardening. This requires a complete review of the system to ensure it can function not only properly but securely. This final pre-operational hardening of the server should include the tightening of IPTables, SELinux and associated services.

Remember that before a system will be accepted by an organization's leadership its configuration will have to be certified. When management is willing to accept the risks the device may create it will be accredited for operation. It only behooves the development team to do a final review and tightening the security of the device or devices as much as possible. This example provides a workable product that can be tailored to an organization and reduces the exposure footprint of potential vulnerabilities.

4. Configure GroundWork Open Source

This is a brief overview of how to configure GWOS for the capabilities this example requires. GWOS has a number of abilities that are not highlighted within this document. As such, proper training and experience are necessary to get the most out of the product.

The GWOS server monitoring interface needs some tailoring to enable parsing of log data for signatures. Five steps are required to enable the GWOS server to function as intended. The Audit server has to be defined as a host that GWOS can monitor. A generic command or commands needs to be generated. This command or commands will be the basis of what future services are created from. These services are what will actually parse the Audit server for specific events of interest. A graphical image must be created to provide a human interface for the Auditable events. The image will be populated with blips that represent services in a "Context Sensitive" fashion. This map will be a graphical representation of the events as they happen to the observer of the system.

Once these steps are finished non-technical users could flag incident response teams as events happen. Additionally, new events can be added as desired by cloning previous services or commands and then tailoring them to new signatures. The use of cloning services greatly speeds the deployment of a new capability. The example below shows that after setting up the first few services, the system is extremely easy to update and modify to new requirements by use of cloning. Before the GWOS server can be configured to monitor the Audit server it has to be able to access it. This is done through Secure Shell (SSH) automated login using asymmetric keys.

4.1. Create SSH keys for GroundWork Open Source server user nagios

The GWOS server will login and execute the monitoring script on the Audit server. The GWOS server will use the nagios user account to access the Audit server. This account must have unprompted access to the system(s) it is attempting to access. The only way to securely do this is through the use of asymmetric cryptographic key pairs. The keys will be on the GWOS server under the nagios account and distributed either automatically or manually. A caveat to this is that the nagios account must be present on the remote machines to be able to do this.

To create SSH key pairs, complete the following actions. Log into the GWOS server as the nagios user, this user should already be present on the GWOS server. Run the command 'ssh-keygen' and answer the prompted questions. Today's versions of Linux default to the Rivest, Shamir and Aldeman (RSA) key processing tools. In the past, Digital Signature Algorithm (DSA) was used by default. Both key processing tools are present and either can be used depending on the arguments used. There are three possible types of keys that can be created 'rsa1', 'rsa' and 'dsa'. Running the 'ssh-keygen' tool with '-t' argument followed by the type generates that type of key.

This example uses the default provided, but whatever the organization chooses to use will be fine. If different key types are placed in the 'authorized_keys' file, they will work with no consequences. The system will still operate appropriately despite key differences.

4.2. Creating a nagios account for remote login on the Audit server

Create the nagios account and then use it to build the required files and directories. This way the files and folders are owned by nagios and not root. So once the account is created log in as nagios before creating the next files.

The most important concern when selecting the password for any host(s) the GWOS server accesses is that it is different than the GWOS server nagios password. Otherwise, if a host is compromised and the password of the nagios account is cracked, it would be possible to login to the GWOS server. From the GWOS server, a malicious attacker could jump to any host on the network that had a nagios account on it.

The recommended practice would be to separate the hosts into sensitivity levels. If GWOS accessed a group of standard workstations, the passwords for the nagios account could be the same on the workstations but not on anything else. As such, passwords would be different on

the devices that needed to be more secure than others with the same account. Workstations would use one nagios password, standard file servers another and more sensitive appliances yet another. The optimal solution would be to have every single system use a different password for the nagios account. When the number of servers the organization maintains reaches a clipping point, this quickly fails. Instead, sensitivity groups are a more practical solution that still meets the intent of the security solution.

Consider that if a malicious hacker can break into a standard desktop, it is more than likely he or she already can do the same to other like desktops very quickly. This is because most desktops are deployed from a standard image. At the same time, a malicious hacker may not be able to compromise a different system, such as a server, with the desktop information that had already been found. This of course depends on what other vulnerabilities are present. Most servers, to contrast desktops, are built for a specific purpose and bastioned, or at least they should be.

This concept holds true in operating environments when desktop and server administrators usually have different Administrative passwords. The accounts may have the same name, yet the Administrative passwords between these different devices usually are not the same. These same concepts should be applied to the nagios accounts throughout the enterprise. The requirements of the nagios account are a little different than a standard account.

The login banner must be suppressed when the nagios account automatically logins in. Otherwise, the automated scripts may read the Message of the Day (MOTD) banner as an error. This error would be presented as a false positive to any reviewer of audit logs on the Audit map. The commands below suppress login banners and the MOTD so nagios does not present them as an error.

```
touch /home/nagios/.hushlogin
chmod 644 /home/nagios/.hushlogin
```

A '.bash_logout' file is required to invoke specific commands for the user when he or she logs out of the system. With the GWOS system, this file will clear the terminal screen which will prevent ghost data from influencing future checks. A simple reference to the full path of the

clear command will accomplish this, as shown below.

```
touch /home/nagios/.bash_logout
chmod 644 /home/nagios/.bash_logout
echo "/usr/bin/clear" >> /home/nagios/.bash_logout
```

A '.bash_profile' file loads all aliases and functions that will be used by the account. In addition, the file sets the environmental variables and startup programs for the account. With the nagios account it is a standard '.bash_profile' that references '.bashrc' and a path updated showing the location of the GWOS scripts folder.

Creating the '.bash_profile':

```
touch /home/nagios/.bash_profile
chmod 644 /home/nagios/.bash_profile
vim /home/nagios/.bash_profile
```

Add the following and write it:

```
# .bash_profile
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        .~/.bashrc
fi
# User specific environment and startup programs
PATH=$PATH:$HOME/libexec
```

The '.bashrc' file is used for when nagios starts up a subshell. The nagios account will implement all commands within the '.bashrc' when the subshell is executed. This allows the commands run during initial login and when starting a new subshell, which in this example references '/etc/bashrc'.

Create the '.bashrc' file:

```
touch /home/nagios/.bashrc
chmod 644 /home/nagios/.bashrc
vim /home/nagios/.bashrc
```

Add the following and write the '~/.bashrc' file:

```
if [ -f /etc/bashrc ]; then
        ./etc/bashrc
fi
```

4.2.1. Configure the SSH communication

The most important group of files in the nagios home directory resides in '.ssh' directory:

```
mkdir /home/nagios/.ssh
chmod 600 /home/nagios/.ssh
```

Within the '~/.ssh' directory there are three files that need to be present. The three files are 'config', 'authorized_keys' and the 'known_hosts' file. These files allow the nagios account the ability to login in to the Audit server from the GWOS server without having to provide a password and help protect against MiTM attacks. These files allow GWOS the ability to use one line SSH encapsulated command that creates a simple Standard Out (STDOUT) return of data. Once that data is returned or the session timer expires, the session ends. The file 'authorized_keys' is where the public key that was generated on the GWOS server as the Nagios user account will be placed.

Generating the 'authorized_keys' file:

```
touch /home/nagios/.ssh/authorized_keys
chmod 600 /home/nagios/.ssh/authorized_keys
```

There are two ways to copy the contents of the SSH keys from the GWOS server to the Audit server. The first way is to login into the GWOS server as the nagios user and copy the contents of the '/home/nagios/.ssh/id_rsa.pub', or the '/home/nagios/.ssh/id_dsa.pub' depending on the key type that was chosen. Then place the contents in the '/home/Nagios/.ssh/authorized_keys' file on the Audit server. The second way possible is if the 'ssh-copy-id' command is available just run it identifying the public key, the target account and machine.

An example of using the 'ssh-copy-id' command from the GWOS server:

```
ssh-copy-id -i /home/nagios/.ssh/<public key file> nagios@192.168.128.9
```

The key should only come from the GWOS server to the monitored device, not the other way around. Enabling two way accesses from monitored devices creates a potentially devastating vulnerability. This would enable a compromised device that is being monitored, direct access to the GWOS server without a password. From that location anything monitored in the site(s) can be crawled.

Now that the key has been distributed to the Audit server, a file is needed to list the hosts that the Audit server has accessed in the past. This is used as a measure to protect against MiTM attacks (Skoudis & Liston, 2006, p. 461). This only affects the host that the file is resident on. The client ensures the authenticity of the target hosts it has logged into by a system specific SSH public key stored within this file (Nemeth et. al., 2007, p. 697).

Create the '/home/nagios/.ssh/known_hosts' file and modify the access permissions:

```
touch /home/nagios/.ssh/known_hosts
chmod 600 /home/nagios/.ssh/known_hosts
```

On the Audit server in the '/home/nagios/.ssh/config' file, there is a list. This list provides the preferred method of authentication and what level of logging should be achieved.

To set this up properly on the Audit server ensure that the authentication is hostbased, then public key and finally password with logging set to quiet.

Create the '/home/nagios/.ssh/config' file:

```
touch /home/nagios/.ssh/config
chmod 600 /home/nagios/.ssh/config
```

Add the following to '/home/nagios/.ssh/config' and write the file:

```
PreferredAuthentications hostbased, publickey, password
LogLevel quiet
```

4.2.2. Configure nagios log check script

A file that temporarily stores seek data used during checks of logs is needed in the home directory of nagios account on the Audit server. This file will be used as a place holder file for the '/usr/local/groundwork/nagios/libexec/check_log2.pl' script. The file will hold byte mark

values as the logs are parsed for patterns. This prevents the rereading of already processed data; as mentioned in the section 3.1.6 this file is zeroed out when the 'log_archiver' script is run.

The file has to be present for the 'check_log2.pl' script to work so use the following:

```
touch /home/nagios/temp
chmod 664 /home/nagios/temp
```

The scripts that the GWOS server normally uses are located on the GWOS server itself. When using remote execution of these scripts, they have to be moved to the target server. To do this, the files must be copied from the '/usr/local/groundwork/nagios/libexec/' directory on the GWOS server to the nagios account on the Audit server. Place the script directory in the home directory of nagios and call it 'libexec.' The purpose of this is to reduce the amount of time it takes to get the results of an executed script.

Copy the 'libexec' folder to the Audit server and adjust the permissions:

```
scp -r /usr/local/groundwork/nagios/libexec nagios@audit.example.com:~
chmod 775 /home/nagios/libexec
```

The files within the '/home/nagios/libexec' directory need to be milled down to only the files that are required for this particular server. Additional files that are left in this directory create possible vulnerabilities on the Audit server. Since this system is being setup to be bastioned, limits must be set on what needs to be on it. The only script that must be on the Audit server for this design is the 'check_log2.pl', everything else is optional.

> As a note the scripts written by many of the contributors for GWOS do not put security at the forefront of the design, instead monitoring capability is. As such any scripts added to the GWOS server or target client must be reviewed by competent programmers with an understanding of secure coding. It would be a shame to create a great tool for log monitoring only to install a security hole.

Since the GWOS server needs to access a new server, and it does not actually know the signature of the server, the signature needs to be added to the 'known_hosts' file. The best way to accomplish this is to copy the data to a Write Once Read Many (WORM) piece of media such as a Compact Disk - Record (CD-R) and paste it directly into the file and then store the key

offline in a safe. This reduces exposure to potential MiTM based attacks and provides a secured reference in case a compromise does happen.

If that is not feasible, then login into the GWOS server as nagios and then SSH from the GWOS server as nagios to the Audit server. The system will state that "The authenticity of host '192.168.128.9 (192.168.128.9)' can't be established. RSA key fingerprint is (16 hexadecimal pairs). Are you sure you want to continue connecting (yes/no)?" Answer 'yes' as the machine is asking if you wish to trust this machine and then adds it to the 'known_hosts' file on the GWOS server.

If, in the future, GWOS hangs or fails when attempting to access the Audit server, while the Audit server is known to be up, then the 'known_hosts' file has been corrupted. The first thing that should be done is verify that a MiTM based attack is not in progress nor has the data been compromised. Once that has been validated the solution would be to either wipe out the entire file or the specific entry has to be removed for that server has to be removed. Now logout of the Audit server and log back in as nagios again. GWOS should be able to access the server now without requiring specific authorization prompts.

The second login should allow you unfettered access to the nagios account on the Audit server. If it does not, there are problems either with the 'known_hosts' file, 'authorized_keys' file, and/or permissions. Go back and check each one and verify that all the settings are correct. The most common causes are either 600 class permissions are not set on the '.ssh' directory on the target and its files or the 'known_hosts' file on the GWOS server is wrong.

As a note, if a password expires on a nagios account, the GWOS server will no longer be able to access the remote system. This is despite the fact that the GWOS will be using key pairs to access the server. Due to this, the password can either be regularly changed which would be the best practice or the password can be set to never expire. Since the accounts for nagios are being setup to work with key pairs there should be no problem in changing the passwords regularly. This can be done with a simple "For Loop", a "Host List" and "Expect plug-in" with a standard scripting language like Perl or Python.

4.3. Create GroundWork Open Source Audit Server Host

It is now time to create the host profile of the Audit server. This is a theoretical representation of the Audit server. This Audit server host will be used to visualize specific events or services checks as they happen.

To start, click on the GWOS icon in the top left corner of the web interface and select 'Configuration'. Now select the 'Hosts' option from the top link on the web interface and then click 'Host wizard' on the left menu. This will allow the administrator to generate a new host by updating a few fields.

The 'New Host Wizard' provides four fields that make up the 'Host Vitals'. The 'Host name' the 'Alias' the 'Address' and the 'Host profile.' Fill in the 'Host name' with the actual host name of the host and the 'Alias' with the FQDN of the host. The 'Address' should be the IP address of the host and the 'Host profile' can be left blank, then press the 'Next' button.

From the 'Host template' drop down, select 'generic-host'. Then, depending on the environment that is setup, the parent devices it belongs to can be selected if there is one. Select 'Next' again and configure what 'Hostgroups' the Audit server will be a part of. This is completely dependent on the environment and it will dictate the way the host appears on the configuration windows in the future. Leave all other fields on this window blank and select 'Next' again.

Now, this window is where the device's services are configured, if there are services already built. If the services that are going to be loaded to the server have already been created, they can be loaded now. Since no services have been created, leave the 'Select service profile' blank and do not select any other services, then click 'Next'. This now gives the option to save the host as a profile to be used later to build other hosts or to continue and create the host. Since only the Audit server is the only item being created right now, click continue.

If the host was not made part of a 'Host Group', it will appear under the 'unassigned' 'Host Group'. As a safety precaution, any changes like adding hosts and implementing multiple services should be saved and committed prior to making additional changes. Be cautious when doing this work as any changes other administrators may have made can affect the GWOS server. To mitigate this risk, a solid change control process on this monitoring solution should be put in place.

First the administrator should test to see if any of the changes made will affect the current environment negatively. This test is called a 'Preflight test' and can be found under the 'Control' link on the top of the page. If the test comes back successful, the changes can be implemented with 'Commit' link under the same position. With all of this done the basic host profile for the Audit server has been completed. Now the host just needs some customization.

Whichever 'Host Group' the Audit server was assigned to needs to be opened by either clicking on it or the '+' beside it. Then click on the 'Host Profile' within the 'Host Group' of the Audit server. This selects the Audit server 'Host Profile' so that it can be updated and edited. The first tab of the host is called the 'Host Detail' which provides the basic configuration options. First ensure that the 'Check command' has the 'check-host-alive' option selected from the drop down and make sure the 'Max check attempts' is set to '3'. This will set the maximum number of checks for a host if it returns anything other than 'Ok'. This is important because it prevents fluke returns from a single point in time. In other words if it GWOS says something is wrong it checked its own homework three times before sending the message.

Next, the 'Notification interval' is an option that is used to determine how often it updates the status of the host. If the status of the host has not changed and it has not reached the 'Notification Interval' the host will not alert again. If the host has reached or passed the 'Notification Interval' and the problem is not fixed, then the system will again alert on the event. In relation to this is the 'Notification period' which should always be set to '24x7.'

The item on this page that that must be addressed is the 'Contact groups.' If this option is used the system administrators, the security administrators and the IA office should all be included here. This will ensure prompt notification of serious events by whatever method the organization has chosen. Examples of this include Short Message Service (SMS), email or even an archaic page. A number of plug-able scripts are available to have the GWOS server enact these notifications directly.

Now that the basics of the Host Profile have been configured, services can be loaded to it. Since the host needs to be tested at this point to ensure it works, at least one service needs to be loaded. Under the 'Services' tab is a selectable list of services, from this scroll list select the 'icmp_ping_alive' and click the 'Add Service(s)' button.

This service will have the GWOS server ping the Audit server equal to the 'Notification check interval' that has been selected for it. A benefit of this test is that it will stop administrators from troubleshooting a problem in the possible wrong direction. This is often the case when more complex service(s) are created. A failed service in a point in time could be just a loss in synchronization or constrained Operating System (OS) service(s). The inclusion of this service will prove that the GWOS server can communicate with the Audit server. To enable these changes, go to the 'Control' section of the 'Configuration' page and do a 'Preflight test.' Ensure the 'Preflight test' succeeds and press 'Commit' and verify the success.

The 'icmp_ping_alive' notification is known as a positive acknowledgement (pos-ack) or it says "Yes I was successful". This is opposed to the rest of checks the Audit server will have added to it which are known as negative acknowledgement (neg-ack) or they say "Yes something is wrong". This difference means that pos-acks flag if they are unable to complete their checks. The neg-acks flag if they complete their check and find something else besides 'Ok', then they respond with an error or unknown. This is not only important from an availability stand point, but could also prevent the insertion of a malicious protocol analyzer. If a malicious user removes the connection between the Audit server and the switch, they could install a protocol analyzer between the two.

Once the device is brought online, the connection would be reestablished as though it was only a slight hiccup in traffic. This may only cause a few neg-ack checks to flag unknown, which may happen regularly due to traffic anyway. Now the aggressor could pull data off the network all day long and use it to attempt a compromise of other hosts. The Audit server is an especially rich target because of all the devices that send information to it.

Unencrypted data sniffed could provide a map of the entire network. This is part of why traffic should be encrypted between the clients and the Audit server including syslog data. The final link between the switch and the Audit server is not the only locations a sniffer could be placed. Anywhere communication between end devices a protocol analyzer insertion can take place. Port security and physical security can help to reduce the chance this problem, but an extra layer of defense is the pos-ack check. As such, if an 'icmp_ping_alive' fails on a critical resource, inspection should immediately follow on all the physical devices in that information chain.

The ‘icmp_ping_alive’ check is the first of many services to be added to the host profile and it is the most important. If a ping fails, GWOS will flag the check, where as if the parsing of log signatures fails to find something it will not flag. In theory, if there was no ‘icmp_ping_alive’ check, the Audit server could be offline and GWOS would only flag as unknown. Until a large amount of “unknowns” have been flagged, the GWOS server may not be inspected and by then it could already be too late. So in essence, the network could be attacked and the GWOS server would show green in many regards. If the GWOS server is getting overloaded with checks, do not remove the ‘icmp_ping_alive’ check, unless there is another service providing pos-acks. Instead, determine where the performance degradation is coming from and resolve it.

The Audit server host profile within GWOS is now complete and additional services can be added as they are created. These service checks can be mapped to devices on a visual map to present a conceptual image of security once this is done. To create a service though, a standard command with replaceable variables has to be generated.

4.4. Create GroundWork Open Source Commands

The GWOS commands are a customized template created by the GWOS administrator. This template is what is used to build up the services on the host, in this case the Audit server. The commands being built require only the ‘check_log2.pl’ script to be located in the host’s ‘/home/nagios/libexec’ directory. On the actual GWOS server the ‘/usr/local/groundwork/nagios/libexec/check_by_ssh’ script will be used to receive the results of the ‘/home/nagios/check_log2.pl’ script from the Audit server.

To begin, login to the GWOS server web interface and click the GWOS icon in the top left corner. Select the ‘Configuration’ option and the ‘Commands’ link on the top of the page. From here the new command will be generated by clicking the ‘New’ link on the left side. The ‘Command Wizard’ will pop-up and ask the Administrator to ‘Select resource macro:’. Check mark the ‘/usr/local/groundwork/nagios/libexec’ option which should default to the variable ‘user1’ then click ‘Next>>‘. If this variable is not available it can be replaced with the equivalent information that it contains. From here the ‘Command Wizard will allow the command to actually be built.

The new window will contain fields for ‘Command name:’, ‘Type’ and ‘Command line:’ configuration portions. There is also a test field to test the command if desired with the fields ‘Host:’, ‘Arguments:’ and ‘Service description:’. The ‘Command name:’ will represent the script that is being used on the Audit server ‘check_log2.pl’ so it will be called ‘check_log2_pl’. The ‘Type:’ is a ‘check’ which is selected from the drop down menu.

The ‘Command line:’ will be where most of the work for the command is done. The command example being used has six variables that are being used in addition to ‘$USER1$’ which represented ‘/usr/local/groundwork/nagios/libexec’. The first variable is ‘$USER17$’ which represents the username ‘nagios’. The second variable is ‘$USER22$’ which represents the directory on the target machine that the ‘check_log2.pl’ is located in, which is ‘libexec’. If these two variables are not defined, utilize the values instead. If they are defined as a different variable, use the other variable name. The other four variables will be defined in the service check.

The command when finished should read ‘$USER1$/check_by_ssh -H $HOSTADDRESS$ -t 60 -l “$USER17$” -C “$USER22 /check_log2.pl -l $ARG1$ -s $ARG2$ -p $ARG3$” ‘ when using the variables. This command uses the ‘/usr/local/groundwork/nagios/libexec/check_by_ssh’ script to access the target host ‘$HOSTADDRESS$’. The connection will use the username ‘nagios’ and only hold the session open for 60 seconds while the ‘/home/nagios/libexec/check_log2.pl’ is run. This script will check the target log directory for a defined Regular Expression (RegEx) while bookmarking its progress with the ‘/home/nagios/temp’ file. The information is returned to the STDOUT of the Audit server. This information is picked up by the ‘/usr/local/groundwork/nagios/libexec/check_by_ssh’ script and passed to the GWOS server. The GWOS server flags the return depending on the results provided by the check on the remote host. Knowing this information, the system can be tested with a known signature before creating the services.

In the testing section of the ‘Command Wizard’ set the ‘Host:’ field to ‘192.168.128.9’. In the ‘Arguments:’ field set it to ‘/var/log/systems/system_name!/home/nagios/temp!root’. Notice that each of the argument values is separated by no space and only an ‘!’. Click the ‘Test’ button and see the return in the field next to the button. The information should populate with all

the values that were set with an exit status. The status should be either ‘0’ or ‘1’ and a description or ‘2’ or ‘255’ and no description. A ‘0’ means nothing was found that matched the pattern for this check. The results would be green and the message returned would be “Ok - No matches found”. A ‘1’ means that something was found and the description presented says what it was. A ‘2’ shows that the command was improperly formatted or the script is missing somewhere. Finally a ‘255’ means either the target could not be found, accessed or is down.

With this information, the command has been created and is ready to be turned into multiple services that will be loaded against the Audit server. Click the ‘Add’ button and ensure that the new command has been added, this is done by closing the tree on the left side and re-opening it. Then click on ‘Control’ on the top bar and click ‘Preflight test’ to ensure the changes made will be successfully updated. With the return of a successful test click the ‘Commit’ link and await a ‘success’. If a success is returned move on to creating a service out of the command that was just created.

4.5. Create GroundWork Open Source Services

The command template has been created, now the ‘services’ that will represent each event signatures for the ‘/var/log/systems/<system_name>‘ file(s) must be developed. This is a time consuming and laborious process as each log location may require the same signature definition. It is best to already have known signatures defined and used those definitions to create RegEx for use in the services. A number of examples will be provided here, but these are just a baseline. The signatures will become more dependent on both the systems at each location, the threats it faces, the operating system types and the software or services that run on them.

The first step is to create a single service check from which other services can be quickly cloned. First, click the GWOS login to the GWOS web interface and click the GWOS icon in the top left corner and then select ‘Configuration’. Click ‘New service’ and the ‘New Service’ window will pop-up. Within the ‘Service name’, type the ‘check_log_system_name_signature_name’. The idea is to make a human readable alert, which an observer can identify when multiple indicators are present on the GWOS dashboard. From here click the ‘Service template’ drop down and select ‘generic-service’ then click ‘Add’.

The new ‘Manage Service’ window will appear which looks like the ‘Host Profile’ window. This location is the where the ‘Service Detail’ is defined. On the first screen, the ‘Normal check Interval’ is what is set for how often the GWOS server will check for this signature on the Audit server. The default for this check is 10 minutes and can be adjusted as needed.

The ‘Normal check Interval’ has to be adjusted for each location based on a few factors. How busy or constrained the CPU(s), Memory and NICs of the server are. Additional factors are how busy the GWOS server is natively and how many SSH sessions the server can handle at one time. Testing and adjusting this will have to be done on a per organization basis. The adjustment and modification of the ‘/etc/ssh/sshd_config’ and ‘/etc/ssh/ssh_config’ files on the Audit server and GWOS server respectively, can throttle the session numbers as necessary.

This will ensure that the system can handle enough concurrent SSH sessions and to prevent the checks from timing out or failing inexplicable. Depending on the security requirements of the organization and how close to real time the organization needs to be, this setting will have to be adjusted. So the factors that determine the ‘Notification check interval’ will depend on the hardware limitations and security requirements. These requirements apply to both the Audit server and the GWOS server and the connections between them.

The ‘Notification interval’ is how often GWOS should re-alert a signature if it is still not green, the default is 60 seconds. This is the second item that needs to be adjusted for the service and should be set in accordance to the organization’s needs. In testing, the default is plenty and should not need to be adjusted for standard services. For the Audit server log checks, this should only execute once so set this value to ‘0’. When a log alert is triggered, it is not going to clear out because it was fixed. Instead the alert will be acknowledged and the follow-up actions will be done by the security administrators and IA office.

Within the ‘Service Check’ tab is the actual service that will be defined from the command created earlier. In the ‘Check command:’ drop down select the ‘check_log2_pl’ command. This will populate the ‘Command definition:’ that was created in the section 2.2.4. In the ‘Command line:’ location will be the name of the script followed by a ‘!’ then ‘ARG1’ then ‘!’ then ‘ARG2’ then ‘!’ and finally ‘ARG3’. The only things that will be replaced are the ‘ARG#’ items with the appropriate values.

Replace ‘ARG1’ with the ‘/var/log/systems/system_name’ that will be checked. Overwrite ‘ARG2’ with ‘/home/nagios/temp’ and ‘ARG3’ with the pattern of the signature that is being looked for. Now in the ‘Host:’ field put in the Audit servers IP address ‘192.168.128.9’ and click the ‘Test’ button. The same return values seen in the command section will be seen here. When everything works in the test run, save the ‘Service Check’ by pressing the ‘Save’ button. Commit the changes made with a standard ‘Pre-flight test’ and ‘Commit’ loop through the ‘Control’ screen.

Now that a single ‘Service Check’ has been created, all future ones can be cloned from it. Under the ‘Service’ configuration location click ‘Clone service’. In the new window, click the ‘Select service’ drop down and click ‘Next>>‘. Create a new ‘Service name’ and click ‘Next>>‘ which will create the new cloned service. Adjust the ‘Normal check interval’ and ‘Notification interval’ under the ‘Service Detail’. Then setup the location and signatures being checked under the ‘Service Check’ ‘Command line:’. Make sure to test all new ‘Service Checks’ before saving them.

Multiple services can be setup before doing a standard commit operation, just be sure to save the changes before changing screens or else they will be lost. It is also not suggested to create more than ten ‘Service Checks’ without committing them. It is really easy to lose all that work with a server lock up or a session failure. Now that creating a service and cloning it have been covered, below is a listing of signatures to help get an organization started.

Service Check Signature Table:

Signature Looks For:	Signature:
Linux Authentication failures	Authentication
Linux Remote Desktop Attempt	rdesktop
Linux Virtual Network Connection	vnc
Windows Disabled Account Login Attempt	"531\s"
Windows Expired Account Login Attempt	"532\s"
Windows Login Attempt Failure	"529\s"
Signature Looks For:	Signature:
Windows Login Attempt Failure "UNK"	"537\s"
Windows Login Attempt on Locked Acct	"539\s"
Windows Login Attempt on Expired Pass	"535\s"
Windows Login Not Allowable	"534\s"
Windows Unauthorized Login	"533\s"
Blocked Access to Audit Server	IPTables

As a note, the IPTables signature only works if a rule for IPTables is configured to send an event to '/var/log/messages'. More information on this subject can be found online. A tutorial for flagging these events can be found in Appendix B of this document under "25 Most Frequently Used Linux IPTables Rules Examples".

4.6. Load Services to Audit Server

The Audit server has been created, a command template was built and services built off that command template. Now it is time to populate the host with the services. Login to the GWOS web interface and click the GWOS icon in the upper left corner then select 'Configuration'. This will automatically send the administrator to the 'Hosts' screen. Click the 'Hosts' tree and select the Audit server 'Host Profile' that was created previously. Select the 'Detail' icon under the server and the 'Services' tab under the profile.

At the bottom of the 'Services' tab on the 'Host Profile' page is the services selection box. Here all the services that have been created are present. All a user has to do is select each

service in the box and click the 'Add Service(s)' button. The service is now loaded to the Audit server. Click back on the 'Host Detail' tab and save all the work done. Now complete the standard 'Preflight test' then 'Commit' loop. The Audit server now has all the newly created services loaded to it.

Any signatures that flag will appear in the services failure section of the GWOS 'Dashboard'. This is a start, but a graphical representation would be a much better way to alert on an event. To do this the services need to be loaded to a map which graphically represents the system(s) to its observers.

4.7. Create the Map for GroundWork Open Source

The image that will be used to load the services to is called a background. When the services have been defined on the background it is then called a map. This is important to understand because this is how GWOS interprets and saves the images.

Any standard graphical program can be used to generate the Portable Network Graphics (png) images. Software such as VISIO which must be paid for, DIA which is free under Gnu's Not Unix (GNU) Pulic License (GPL) or even paint can do the job. The image(s) need to be named with no special characters or spaces in the name except for underscore(s) as desired. After the image is created with a suggested size of about 800x600 Dots Per Inch (dpi) it can be uploaded into the GWOS server.

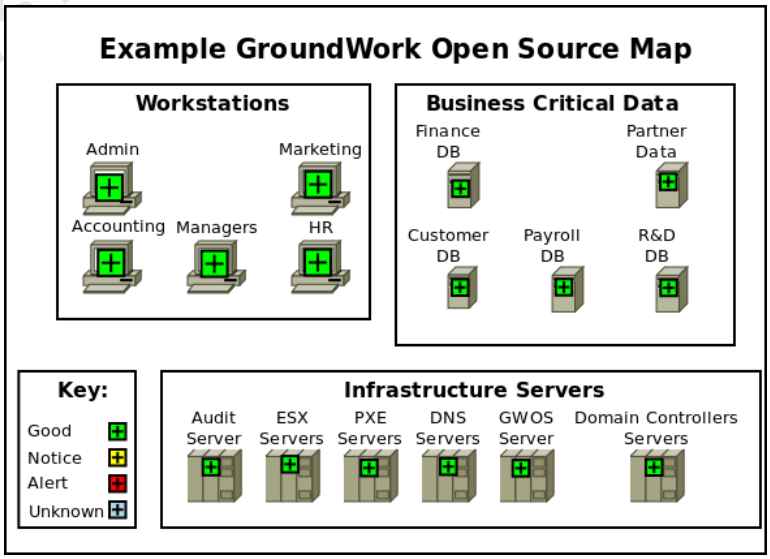
The background image should have an image or icon that represents each system that is defined by the '/var/log/systems/system_name' on the Audit server. This is because the service is dependent on that particular file when it was created. The service check represents an event check on that file for a particular signature. If the icon represents one hardware system or entire business suite it doesn't matter because the alert text will point to the correct system. Once the image is complete it can be loaded into the GWOS server.

4.8. Integrating Maps into NagVis

Nagvis is the graphical interpreter of the data in GWOS used to present the information to the users of the system. To load the background the administrator logs into the GWOS server through the standard homepage. Once there, the administrator clicks the GWOS symbol in the upper left corner and selects 'NagVis Admin' from the drop down. The administrator then right

clicks the background and clicks ‘Manage > Backgrounds’. This brings up the background manager web interface. From there, the administrator goes to the upload background image location and browses to the location of the file and clicks the upload button.

Once the background has been updated it is time to create the NagVis Map which tracked services will be placed upon. The administrator right clicks the background of the NagVis Admin screen again and clicks ‘Manage > Maps’. This ‘Map’ screen will be used to create the new NagVis map, fill in the ‘Map Name’ with no spaces. Then in the “User with read permissions” and the “User with write permissions” fill in ‘EVERYONE’. The ‘EVERYONE’ group is the default choice of GWOS 5 to allow a map to be viewed. Specific groups and views can be created on the GWOS server but this has can lead to problems with testing on GWOS 5.3. Due to these problems, the example only uses the EVERYONE category with the option to tighten security as the administrators choose later on. Choose the “map Iconset” as ‘std_medium’ for best visibility without cluttering the map. Lastly, choose the Background from the drop down that was just uploaded and click the create button. The NagVis Map for the Audit server has just been created. Now the map has to be populated with the data that will alert the watch team. Below is an example map that could be populated with services to signify a event of interest.



Note: GWOS 6 provides a host of new capabilities and controls with user and group views. Since the system administrators should be reviewing system related alerts not Audit events and the security administrators doing the opposite. It is suggested to separate the views of the information that they each party can see. GWOS 6 provides the capability to do this with very specific filters. Additionally, it is the only version that can be purchased currently.

4.9. Place Syslog tracked services on NagVis Maps

Now that the background has been uploaded and the map created, it must be populated with data. Access the NagVis Admin page by login into the GWOS standard web login and clicking the GWOS icon in the top left followed by the 'NagVis Admin' link. Right click on the splash page click 'Open Map' select the number range the map falls under and click the "Map Name". Now that the map is present the services icons can be loaded.

To load a servers to a particular portion of the map right click it select 'Add Object > Icon > Service' and then click on the map. This will bring up the window to add a new service to populate the map at that location. The service can be moved if it is not in the desired location after creation. Select the 'host_name' that represents the 'Host Profile' name of the Audit server. Then select the 'service_description' which represents the name of the service being added to the map and click 'Save'. The icon has been added to the map and can be moved now by clicking, holding and dragging the icon to the desired location.

After adding a service make sure to right click and click the 'Save' option. Do this before adding a new icon else the positioning completed for the previous icon will revert to its creation location. Complete these steps for each one of the services and the map will be fully populated. Now the GWOS Dashboard map is complete and can be viewed by returning to the GWOS homepage.

5. Feeding Multiple Sites

The difference between serving one site or multiple sites with this Auditing solution differs very little. Since the designs of these systems were thought of in a modular capacity, the servers can build on each other. Audit servers can feed other Audit servers and GWOS child servers and agents can push their data from site locations to the enterprise monitoring solution.

5.1. Feeding an Enterprise Audit server

The pushing of data from a standard Audit server at a site level up a hierarchical chain is not complex with syslog based tools. Though it should be considered what form of enterprise level encryption will be used to transfer the data from site to site. Sending data between devices by PKI based encryption is a solid start. Data between sites should be wrapped in an Enterprise class VPN solution.

Only three additional items are needed to transfer the data between the site centralized Audit server and the enterprise Audit server. Encryption between the devices can also be PKI based using the native rsyslog TLS or stunnel SSL/TLS system(s). An adjustment to the security protocols of the servers to enable the transfer of the logs up the hierarchy of sites to like devices. Lastly, an entry at the bottom the '/etc/rsyslog.conf' that points to the enterprise server is required. This configuration is in essence a client talking to another centralized site server.

To add this entry just place a '*.* @@audit.enterprise.com' at the bottom of the '/etc/rsyslog.conf' file and restart the rsyslog service. Once this is done, the logs will transfer to the target server as well as to the local storage. On the enterprise Audit server, the system breakout should represent the hierarchy of the enterprise. So instead of a '/var/log/system/system_name' it should be depicted as '/var/log/system/global_location/system_name' or as '/var/log/system/gobal_location_system_name'. The former requires adjustments to be made to all the breakout scripts, but it is a cleaner way of breaking out the logs at the enterprise level. The latter requires no adjustment to the scripts, but it does require more work to breakout on the Enterprise GWOS maps. After these adjustments have been made, enterprise events can be seen from one location.

5.2. Feeding an Enterprise GroundWork Open Source Server

If the solution is for a multi-site enterprise, the deployment of GWOS is slightly different. GroundWork Monitoring Enterprise uses GroundWork Distributed Monitoring Agents (GDMA) and other child servers to aggregate the information to one point. Child servers can be placed at each site to pull information for that site and present it to the enterprise administrators.

Additionally, the Enterprise GWOS server could review logs on the aggregate log server. These logs could be broken out on the map depicting multiple sites. Then a drill down option

can be present on the map to display the system specific events. This all depends on how the specific logs are broken out and how many sites are within the enterprise. The way the enterprise is laid out is not the only item that dictates the deployment of the GWOS enterprise solution. The number of systems that the GWOS enterprise solution monitors also contributes to the design of the monitored resources.

Each host that the GWOS monitors at each site incurs licensing limitations. The host limit of the license is decremented by one for each host viewed by GWOS. The number of hosts at each site and throughout the enterprise will determine the amount of revenue that the organization is willing to invest in the solution.

The amount of bandwidth between sites and the connection confidentiality of those links will also dictate the information that will be transferred between sites. If the sites are limited in the amount of data that can be sent between each other, putting copious amounts of host information in the pipe could limit the enterprise in other ways. It would be undesirable to have IP addresses, hosts names and status of services transferred without strong encryption methods on the public networks. All these items have to be reviewed during the survey to determine if this course of action is the correct one for the organization.

6. Conclusion

The deployment of a secured Audit server is a solid first step for an organization. That first step in a direction that leads to being prepared for malicious events, disasters and standard system failure. Additionally the organization has begun to prepare forensically and legally for events that will have to be proven in a court of law.

The automation of transfers, encryption and offload prevents missteps along the way and deviations from the process. Those deviations could be called to question in legal situations and end up being challenging to prove in court. Having a solid baseline of data that shows what normal traffic looks like provides a comparison for future events. Those malicious events will happen to the organization. Falling in line with the ostrich complex will not prevent bad things from happening (Waitley, 2012). Instead an organization should start preparing for those events now.

Personnel must review logs on a regular basis, otherwise there is no point in having them. Unfortunately, this is only possible if the logs are small in quantity or if the logs can be presented in a reduced view. Just having the logs pulled and stored does not prove due diligence. Without due diligence the organization's leadership could be found legally liable (Crilly & Sherman, 2010, p. 4). Instead, the organization must not only review the logs but act on suspicious events. These suspicious events can be picked up by GWOS and presented to the security teams.

These suspicious event signatures are reviewed by the security teams along with the associated logs they were found in. The action that follows due to this review is dependent on the nature of the event after the review. Was the signature a true positive, false positive, true negative or false negative. This review information will cause the team to adjust signatures and document the change, act on events or document false positives.

Use of the tools presented here allows the organization the opportunity to maintain their log data in a concise and organized format. Logs are maintained securely in an offline format for the retention period of the organization. Data can be transmitted to the different devices securely in a number of different ways. The tools enable review of the logs securely and in a manner that can be tracked. Event signatures can be acted on in near real time as opposed to months down the road when the residual effects are felt.

The biggest benefit is that use of these tools helps to prove due care and due diligence of the organization's information. Without that proof, the organization and its leadership could be held legally and financial at fault for the results of an event. These tools are not the whole solution for an organization attempting to prove due care and due diligence, but they are a piece of the puzzle. Additionally if these items are implemented the organization is now forensically and legally ready to take care of its data during an event.

An organization must choose its next steps carefully and review what its intentions are. The solutions chosen must fit into is overall Information Security Management System (Vinod, Anoop, Firosh, Sachin, Sangit & Siddharth, 2008, p. 17). This baseline of a secured Audit server with GWOS monitoring is the first step. Tailoring it to the organizations needs is the next and most important step. Deploying the solution and reviewing it regularly as designed to fit the organization's needs is the where this document leaves off and the administrators begin.

7. References

Crilly, W. & Sherman, A. (2010) The AMA Handbook of Due Diligence. New York: AMACON

GroundWork, Inc (n.d). Retrieved August 9, 2012 from http://www.gwos.com/

Harris, S. (2010) CISSP All-in-One Exam Guide (5th ed.). New York: McGraw-Hill Osborne Media

Harper, A., Harris, S., Ness, J., Eagle, C., Lenkey, G. & Williams, T. (2010) Gray Hat Hacking: The Ethical Hackers Handbook (3rd ed.). New York: McGraw-Hill Osborne Media

Kennedy, D., O'Gorman, J., Kearns, D. & Aharoni, M. (2011) Metasploit: The Penetration Tester's Guide. San Francisco: No Starch Press

Skoudis, E. & Liston, T. (2006) Counter Hack Reloaded (2nd ed.). New Jersey:Prentice Hall

Nemeth, E., Snyder, G., Hein, T., McGinley, L., Whaley, B., Boggs, A., …Schweikert, D. (2007) Linux Administration Handbook (2nd Ed.). Upper Saddle River: Pearson Education, Inc.

Nessus Vulnerability Scanner (n.d.). Retrieved August 9, 2012 from http://www.tenable.com/products/nessus

Vulnerability Management Software - Nexpose (n.d.). Retrieved August 9, 2012 from http://www.rapid7.com/products/nexpose/

Tipton, H. (2010) Official (ISC)2® Guide To The CISSP® CBK® (2nd Ed.). Boca Raton: Auerbach Publications Taylor Francis Group, LL

Vinod, V., Anoop, M., Firosh, U., Sachin, S., Sangit, P. & Siddharth (2008) Application Security in the ISO27001 Enviroment. Cambridgeshire: IT Governance Ltd

Waitley, D. (April 18, 2012) Potential for High Achievement. Retrieved April 19, 2012 from http://www.appleseeds.org/waitley_Icarus.htm

Appendix A: Log Management Scripts

Create the system specific listing file at '/root/system':

```
vim /root/system
```

For each system create an entry in the file of the following:

```
system_name:/var/log/systems/system_name
system_ip:/var/log/systems/system_name
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/log_move
chmod 700 /root/log_move
```

Create the system parsing temporary folder at '/root/systems':

```
mkdir /root/systems
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/log_move
chmod 700 /root/log_move
```

Create the system specific log breakout script at '/root/log_move':

```
vim /root/log_move
```

Update the file with the following and write it:

```
#!/bin/bash
#
#Author: Christopher S.  Duffy
#Date: August 12, 2011
#Email: Christopher.S.Duffy@gmail.com
#
#Purpose: The purpose of this script is to read a list of SYSTEMS and
# SYSTEM_LOG_FILES from an outside file.  This script then takes
# /var/log/messages and splits it out into multiple system specific log files.
```

Continued on the next page …

```
# This script is designed to be run from the root crontab.
#
SYSTEMS=`cat /root/system`
NUMBER_OF_SYSTEMS=`echo $SYSTEMS | wc -w`
SOURCE_UNALTERED=/var/log/messages
SOURCE_ALTERED=/root/systems/messages_temp
CURRENT_SYSTEMS_PROCESSED=0
TEMP_FILE="/root/systems/temp"
mv $SOURCE_UNALTERED $SOURCE_ALTERED
touch $SOURCE_UNALTERED
chmod 600 "$SOURCE_UNALTERED"
cat /dev/null > "$SOURCE_UNALTERED"
/etc/init.d/rsyslog restart
for LINE in $SYSTEMS
do
        ((CURRENT_SYSTEMS_PROCESSED++))
        PERCENT=`awk 'BEGIN{printf("%0.2f", \
'$CURRENT_SYSTEMS_PROCESSED'/'$NUMBER_OF_SYSTEMS'*100)}'`
        ITEMS=$( echo $LINE | tr ":" "\n" )
        ITEMS=($ITEMS)
        SYSTEM_NAME=${ITEMS[0]}
        SYSTEM_LOG=${ITEMS[1]}
        echo ""
        echo "Moving logs to system specific log files"
        echo ""
        echo "Number of SYSTEMS processed: $CURRENT_SYSTEMS_PROCESSED \
of $NUMBER_OF_SYSTEMS"
        echo "Completed: $PERCENT%"
```

Continued on the next page ...

```
        echo ""
        echo "The logs for this system $SYSTEM_NAME are being processed"
        echo "This logs are being placed here: $SYSTEM_LOG"
        echo ""
        touch "$SYSTEM_LOG"
        chmod 755 "$SYSTEM_LOG"
        grep "$SYSTEM_NAME" "$SOURCE_ALTERED" >> "$SYSTEM_LOG"
        sed "/$SYSTEM_NAME/d" "$SOURCE_ALTERED" > "$TEMP_FILE"
        mv "$TEMP_FILE" "$SOURCE_ALTERED"
done
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/log_move
chmod 700 /root/log_move
```

Create the log archive script at '/root/log_archiver':

```
vim /root/log_archiver
```

Update the '/root/log_archiver' file with the following:

```
#!/bin/bash
#
#Author: Christopher S.  Duffy
#Date: August 12, 2011
#Email: Christopher.S.Duffy@gmail.com
#
#Purpose: The purpose of this script is to move the logs from
# active view of GROUNDWORK to an archive folder.
# This will prevent extremely large log files from building up.
# If the files get to big GROUNDWORK cannot parse
# through the information quick enough.  This will cause checks
```

Continued on the next page …

```
# to time out in GROUNDWORK and eventually lock up
# the centralized Audit server.
#
LOG_FILE=/var/log/systems/*
LOG_DIR=/var/log/systems
ARCHIVE_DIR=/var/log/archive
TEMP_FILE=/home/nagios/archive_temp
CLEAR=/dev/null
LOG_FILE=($LOG_FILE)
#Pull the log file names from the original log directory
for i in "${!LOG_FILE[@]}"
do
        TEMP=`basename ${LOG_FILE[$i]}`
        declare LOG_FILE[$i]=$TEMP
        echo ${LOG_FILE[$i]}
done
#Move the data from the log files and clear the contents.
for i in "${!LOG_FILE[@]}"
do
        /bin/cat $LOG_DIR/${LOG_FILE[$i]} >> $ARCHIVE_DIR/${LOG_FILE[$i]}
        chmod 755 $ARCHIVE_DIR/${LOG_FILE[$i]}
        /bin/cat $CLEAR > $LOG_DIR/${LOG_FILE[$i]}
done
#Clear the temp file used for parsing during the GROUNDWORK checks.
#This file prevents multiple alerts on the same log entry.
#Since the logs are being moved out of the parsed directory this file
#should be nulled so that it does not get too big.
/bin/cat $CLEAR > $TEMP_FILE
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/log_archiver
chmod 700 /root/log_archiver
```

Create the compression script for the logs at '/root/move_compressed_logs':

```
vim /root/move_compressed_logs
```

Update the file with the following and write it:

```
#!/bin/bash
#
#Author: Christopher S.  Duffy
#Date: August 12, 2011
#Email: Christopher.S.Duffy@gmail.com
#
# Purpose: The purpose of this script is to move the logs from the archive directory in /var
# to a compressed directory in a different partition.  This is done in Three stages:
# First: The logs are moved from the /var/log/archive directory to the
# /var/log/compressed_logs directory
# Second: The logs are compreseed and dated in /var/log/compressed_logs directory
# Third: The logs are moved from the /var/log/compressed_logs directory to the
# /data/compressed_logs directory.
COMPRESSED_FILE=/var/log/compressed_logs/*
COMPRESSED_DIR=/var/log/compressed_logs
LOG_ARCHIVE=/var/log/archive
ARCHIVE_FILES=/var/log/archive/*
COMPRESSED_DATA_DIR=/data/compressed_logs
COMPRESSED_FILE=($COMPRESSED_FILE)
DATE=`date "+%V_WOY_compressed_%B_%d_%Y%n"`
TODAY=`date "+%B_%d_%Y%n"`
if [ "$(ls -A "$LOG_ARCHIVE")" ]; then
```

Continued on the next page …

```
        echo "The Logs are being moved today the $TODAY"
        echo ""
else
        echo "There are no logs to move today the $TODAY"
        echo ""
        exit
fi
#Move to the log archive directory
cd $LOG_ARCHIVE
for FILE in $(find $LOG_ARCHIVE -type f)
do
        #Check to see if a file is zipped, if it is move it to the /data directory.
        #If the file is not zipped gzip it and then move it to the /data directory.
        /bin/gzip -t $FILE 2>/dev/null
        if [ $? -eq 0 ]; then
                TEMP=`basename $FILE`
                mv $FILE $COMPRESSED_DIR/$TEMP
        else
                TEMP=`basename $FILE`
                /bin/tar czf ${FILE}.$DATE.tar.gz $FILE --remove-files
                FILE="$FILE.$DATE.tar.gz"
                TEMP=$TEMP.$DATE.tar.gz
                echo "Modified File $FILE"
                mv $FILE $COMPRESSED_DIR/$TEMP
        fi
done
#Move the compressed log files to the new destination.
for i in "${!COMPRESSED_FILE[@]}"
```

Continued on the next page ...

```
do
        SOURCE=${COMPRESSED_FILE[$i]}
        TEMP=`basename ${COMPRESSED_FILE[$i]}`
        declare COMPRESSED_FILE[$i]=$TEMP
        DESTINATION=$COMPRESSED_DATA_DIR/${COMPRESSED_FILE[$i]}
        mv $SOURCE $DESTINATION
done
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/move_compressed_logs
chmod 700 /root/move_compressed_logs
```

Create the last day of the month calculator at '/root/last_day_of_month':

```
vim /root/last_day_of_month
```

Update the file with the following and write it:

```
#!/bin/bash
#
#Original Author: Scott Rowley
# Modified by: Christopher S.  Duffy
# email: Christopher.S.Duffy@gmail.com
# Purpose: To set the last day of the month on Gregorian Calendar for cron jobs
TODAY=`/bin/date +%d`
TOMORROW=`/bin/date +%d -d "1 day"`
# See if tomorrow's day is less than today's
if [ $TOMORROW -lt $TODAY ]; then
        exit 0
else
        exit 1
fi
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/last_day_of_month
chmod 700 /root/last_day_of_month
```

Create the end of month cleanup script at '/root/end_of_month_consolidation':

```
vim /root/end_of_month_consolidation
```

Update the file with the following and write it:

```
#!/bin/bash
#
#Author: Christopher S.  Duffy
#Date: August 12, 2011
#Email: Christopher.S.Duffy@gmail.com
# Purpose: The purpose of this script is to clean up any logs that
# may not have been organized and processed before the end of the month.
#
# This script is broken into three modules which are portions from
# previous scripts used to control the log functions.
#
# Module: 1
# Purpose: Move the logs from the /var/log/messages file to the appropriate system files
#
# Module 1 Variables
SYSTEMS=`cat /root/system`
NUMBER_OF_SYSTEMS=`echo $SYSTEMS | wc -w`
VAR_LOG_MESSAGES=/var/log/messages
VAR_LOG_MESSAGES_TEMP=/root/systems/messages_temp
CURRENT_SYSTEMS_PROCESSED=0
TEMP_FILE="/root/systems/temp"
#
```

Continued on the next page …

```
# Module 2 Variables
VAR_LOG_SYSTEMS_FILE=/var/log/systems/*
VAR_LOG_SYSTEMS_DIR=/var/log/systems
VAR_LOG_ARCHIVE_DIR=/var/log/archive
TEMP_NAGIOS_FILE=/home/nagios/archive_temp
CLEAR=/dev/null
VAR_LOG_SYSTEMS_FILE=($VAR_LOG_SYSTEMS_FILE)
#
# Module 3 Variables
VAR_LOG_COMPRESSED_FILE=/var/log/compressed_logs/*
VAR_LOG_COMPRESSED_DIR=/var/log/compressed_logs
VAR_LOG_ARCHIVE_DIR=/var/log/archive
VAR_LOG_ARCHIVE_FILES=/var/log/archive/*
DATA_COMPRESSED_DIR=/data/compressed_logs
VAR_LOG_COMPRESSED_FILE=($VAR_LOG_COMPRESSED_FILE)
DATE=`date "+EOM_compressed_%B_%d_%Y%n"`
#
mv $VAR_LOG_MESSAGES $VAR_LOG_MESSAGES_TEMP
touch $VAR_LOG_MESSAGES
chmod 600 "$VAR_LOG_MESSAGES"
cat /dev/null > "$VAR_LOG_MESSAGES"
/etc/init.d/rsyslog restart
for LINE in $SYSTEMS
do
        ((CURRENT_SYSTEMS_PROCESSED++))
        PERCENT=`awk 'BEGIN{printf("%0.2f", \
'$CURRENT_SYSTEMS_PROCESSED'/'$NUMBER_OF_SYSTEMS'*100)}'`
        ITEMS=$( echo $LINE | tr ":" "\n" )
```

Continued on the next page ...

```
        ITEMS=($ITEMS)
        SYSTEM_NAME=${ITEMS[0]}
        SYSTEM_LOG=${ITEMS[1]}
        echo ""
        echo "Moving logs to system specific log files"
        echo ""
        echo "Number of SYSTEMS processed: $CURRENT_SYSTEMS_PROCESSED \
of $NUMBER_OF_SYSTEMS"
        echo "Completed: $PERCENT%"
        echo ""
        echo "The logs for this system $SYSTEM_NAME are being processed"
        echo "This logs are being placed here: $SYSTEM_LOG"
        echo ""
        touch "$SYSTEM_LOG"
        chmod 755 "$SYSTEM_LOG"
        grep "$SYSTEM_NAME" "$VAR_LOG_MESSAGES_TEMP" >> \
"$SYSTEM_LOG"
        sed "/$SYSTEM_NAME/d" "$VAR_LOG_MESSAGES_TEMP" > "$TEMP_FILE"
        mv "$TEMP_FILE" "$VAR_LOG_MESSAGES_TEMP"
done
#
# Module: 2
# Purpose: Archive the logs to the archive directories
#
#Pull the log file names from the original log directory
for i in "${!VAR_LOG_SYSTEMS_FILE[@]}"
do
        TEMP=`basename ${VAR_LOG_SYSTEMS_FILE[$i]}`
        declare LOG_FILE[$i]=$TEMP
```

Continued on next page….

```
        echo ${VAR_LOG_SYSTEMS_FILE[$i]}
done
#Move the data from the log files and clear the contents.
for i in "${!VAR_LOG_SYSTEMS_FILE[@]}"
do
        cat $VAR_LOG_SYSTEMS_DIR/${VAR_LOG_SYSTEMS_FILE[$i]} \
>> $VAR_LOG_ARCHIVE_DIR/${VAR_LOG_SYSTEMS_FILE[$i]}
        chmod 755 $VAR_LOG_ARCHIVE_DIR/${VAR_LOG_SYSTEMS_FILE[$i]}
        cat $CLEAR > $VAR_LOG_SYSTEMS_DIR/${VAR_LOG_SYSTEMS_FILE[$i]}
done
#Clear the temp file used for parsing during the GROUNDWORK checks.  This file
# prevents multiple alerts on the same log entry.  Since the logs are being moved out of the #
parsed directory this file should be nulled so that it does not get too big.
/bin/cat $CLEAR > $TEMP_NAGIOS_FILE
#
# Module: 3
# Purpose: Compress the logs
#
if [ "$(ls -A "$VAR_LOG_ARCHIVE_DIR")" ];then
        echo "The Logs are being moved today the $TODAY"
        echo ""
else
        echo "There are no logs to move today the $TODAY"
        echo ""
        exit
fi
#Move to the log archive directory
cd $VAR_LOG_ARCHIVE_DIR
for FILE in $(find $VAR_LOG_ARCHIVE_DIR -type f)
```

Continued on next page….

```
do
# Check to see if a file is zipped, if it is move it to the /data directory.
# If the file is not zipped gzip it and then move it to the /data directory.
        gzip -t $FILE 2>/dev/null
        if [ $? -eq 0 ]; then
                TEMP=`basename $FILE`
                mv $FILE $VAR_LOG_COMPRESSED_DIR/$TEMP
        else
                TEMP=`basename $FILE`
                tar czf ${FILE}.$DATE.tar.gz $FILE --remove-files
                FILE="$FILE.$DATE.tar.gz"
                TEMP=$TEMP.$DATE.tar.gz
                echo "Modified File $FILE"
                mv $FILE $VAR_LOG_COMPRESSED_DIR/$TEMP
        fi
done
#Move the compressed log files to the new destination.
for i in "${!VAR_LOG_COMPRESSED_FILE[@]}"
do
        SOURCE=${VAR_LOG_COMPRESSED_FILE[$i]}
        TEMP=`basename ${VAR_LOG_COMPRESSED_FILE[$i]}`
        declare VAR_LOG_COMPRESSED_FILE[$i]=$TEMP
        DESTINATION=$DATA_COMPRESSED_DIR/\
${VAR_LOG_COMPRESSED_FILE[$i]}
        mv $SOURCE $DESTINATION
done
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/end_of_month_consolidation
chmod 700 /root/end_of_month_consolidation
```

Create the email for log offload request at '/root/offload_request_mail':

```
vim /root/offload_request_mail
```

Update the file with the following and write it:

```
ALCON,

This is a reminder that the logs on Example.com Audit server need to be burned to a disk.

The procedure for accomplishing this is:
- Obtain the rack keys to physically access the Audit server rack.
- Obtain from the Information Assurance office a DVD burner and a blank CD-R or DVD-R.
- Attach the burner to the Audit server and insert the disk.
- Login to the Audit server as yourself.
- Run the command 'sudo /data/burn_to_disk' to successful completion.
- Eject the disk from the drive and detach the drive from the Audit server.
- Return the drive to IA and have them add the disk to the secure media log.
- Place the disk in the Audit safe and add it to the safe inventory.
- Add the disk to the Organizations file plan.  Have a great day!

//SIGNED//
Example.com Audit Server
Example Organization
Anywhere USA
Problems administrators@example.com
Phone: 555-5555
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/offload_request_mail
chmod 700 /root/offload_request_mail
```

Create the email for completion of log offload at '/root/offload_complete_mail':

```
vim /root/offload_complete_mail
```

Update the file with the following and write it:

```
ALCON,

- This is a notification that the logs on the Audit server were burned to a disk.
- Please ensure the disk was added to IA's Media Log.
- Please ensure the disk as added to the Audit safe and the safe inventory.
- Please add the disk to the Organization‟s file plan.
- The offloaded logs disk will be stored in the Audit safe.
- If you have any questions, please contact <Backup Office>.  Have a great day!

//SIGNED//
Example.com Audit Server
Example Organization
Anywhere USA
Problems administrators@example.com
Phone: 555-5555
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/offload_complete_mail
chmod 700 /root/offload_complete_mail
```

Create the email for completion of log offload at ‘/root/failed_offload_mail’:

```
vim /root/failed_offload_mail
```

Update the file with the following and write it:

```
ALCON,

- This is a notification that the logs on the Audit server failed to to a disk.
- Please attempt to troubleshoot the problem and resolve it at your earliest convenience.
Have a great day!
```

Continued on next page….

```
//SIGNED//
Example.com Audit Server
Example Organization
Anywhere USA
Problems administrators@example.com
Phone: 555-5555
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/failed_offload_mail
chmod 700 /root/failed_offload_mail
```

Create the '/root/encrypt_move_mail_logs' script:

```
vim /root/encrypt_move_mail_logs
```

After adding the following to '/root/encrypt_move_mail_logs' write the script:

```
#!/bin/bash
#
#Author: Christopher S.  Duffy
#Date: August 12, 2011
#Email: Christopher.S.Duffy@gmail.com
#
#Purpose: The purpose of this script is to email the administrators
# and security personnel when logs need to be burned to a disk.
#
COMPRESSED_FILE=/data/compressed_logs/*.gpg
COMPRESSED_DIR=/data/compressed_logs
COMPRESSED_FILE=($COMPRESSED_FILE)
FILES_TO_MOVE=/data/compressed_logs/*
FILES_TO_MOVE=($FILES_TO_MOVE)
ENCRYPTED_DIR=/data/encrypted_logs
```

Continued on next page….

```
LAST_MONTH=`date -d '1 month ago' +%B`
DATE=`date -d '1 month ago' +%B_%Y%n`
TODAY=`date "+%B %d %Y%n"`
CONSOLIDATED_LOGS="consolidated_logs.$DATE.tar"
#EMAIL Variables
SUBJECT="Time to backup logs off of the Audit server"
EMAIL_IA="IA@Example.com"
EMAIL_ADMIN="Administrators@Example.com"
EMAILMESSAGE=/root/offload_request_mail
PASSPHRASE=/root/.gnupg/passphrase
if [ "$(ls -A "$COMPRESSED_DIR")" ];then
        echo "The Logs are being offloaded today the $TODAY"
        echo ""
        /bin/mail -s "$SUBJECT" "$EMAIL_IA" < $EMAILMESSAGE
        /bin/mail -s "$SUBJECT" "$EMAIL_ADMIN" < $EMAILMESSAGE
else
        echo "There are no logs to be offloaded today the $TODAY"
        echo ""
        exit
fi
cd $COMPRESSED_DIR
/bin/tar cf $CONSOLIDATED_LOGS *.tar.gz --remove-files
/bin/cat $PASSPHRASE | /usr/bin/gpg --homedir /root/.gnupg --batch --sign --encrypt \
passphrase-fd 0 $COMPRESSED_DIR/$CONSOLIDATED_LOGS
rm -rf $CONSOLIDATED_LOGS
FILES_TO_MOVE=/data/compressed_logs/*
FILES_TO_MOVE=($FILES_TO_MOVE)
for LOGS in "${!FILES_TO_MOVE[@]}"
do
        if [[ ${FILES_TO_MOVE[$LOGS]} == *.gpg ]]; then
```

Continued on next page….

```
                TEMP=`basename ${FILES_TO_MOVE[$LOGS]}`
                mv $COMPRESSED_DIR/$TEMP $ENCRYPTED_DIR/$TEMP
        fi
done
```

Change the ownership and permissions for the script to read, write and execute for root:

```
chown root:root /root/encrypt_move_mail_logs
chmod 700 /root/encrypt_move_mail_logs
```

As a note, the '/root/encrypt_move_mail_logs' script requires Gnu's Not Unix (GNU) Privacy Guard (GPG) for encrypting and digitally signing the logs. Once the keys have been created, a passphrase file is needed to provide 'gpg' the ability to encrypt the logs automatically. The use of a passphrase file prevents the phrase from being present as an argument in a script or cron job and is located in '/root/.gnupg/passphrase' and only allows root to read the file. This is not ideal, but, if access to the server's root account is gained, any logs can be viewed anyway. This just means that all historical logs are also in danger of being compromised in the future. If a compromise is ever found, all keys for the server and its passphrases must be changed when rebuilt. The passphrase will be placed in the root home directory 'gpg' store of the Audit server.

Create the '/root/.gnupg/passphrase' file and add the passphrase:

```
vim /root/.gnupg/passphrase
```

Reduce the exposure of the '/root/.gnupg/passphrase' file:

```
chmod 500 /root/.gnupg
chmod 400 /root/.gnupg/passphrase
```

Create the '/root/backup_mail_logs':

```
vim /root/backup_mail_logs
```

Add the following to '/root/backup_mail_logs' and write the file:

```
#!/bin/bash
#
#Author: Christopher S.  Duffy
#Date: September 1, 2011
#Email: Christopher.S.Duffy@gmail.com
#
#Purpose: The purpose of this script is to mount a USB writer,
# create an ISO from the encrypted files specified and then Burn them to disk.
# The script then emails the administrators and security personnel
# when logs have been burned to a disk.
ENCRYPTED_DIR=/data/encrypted_logs
ENCRYPTED_FILES=/data/encrypted_logs/*
ENCRYPTED_FILES=($ENCRYPTED_FILES)
DATE=`date -d '1 month ago' +%B_%Y%n`
BURNT_NAME=$DATE.logs
ISO_NAME=$DATE.logs.iso
TODAY=`date "+%B %d %Y%n"`
# Find the newest cdrom device which should be the USB drive
MOUNT_DEV=`ls -t /dev/scd* | head -1`
MOUNT=/mnt/burner
#EMAIL Variables
SUBJECT="The logs on the Audit have been burned to Media"
SUBJECT_BAD="The logs on the Audit server failed to burn to Media"
EMAIL_IA="IA@Example.com"
EMAIL_ADMIN="Administrators@Example.com"
EMAILMESSAGE=/root/offload_complete_mail
BADMESSAGE=/root/failed_offload_mail
#Finds the USB burner that has been plugged in and loads the device address into a \
reference variable
```

Continued on next page….

```
DEVICE=`cdrecord -scanbus 2>/dev/null | egrep 'CD-ROM|DVDRW|CD|DVD-ROM' \
| awk '{print $1}'`
#Checks to see if any device was found
if [ -z "$DEVICE" ];then
        echo "No burning device was found please ensure the USB device is attached and \
try again!"
        exit
else
        echo "The USB Burner is attached at dev: $DEVICE"
fi
#Check to see if any files are actual in the encrypted directory
#This is done because of the possibility of multiple administrators sharing the data \
custodian responsibility
if [ "$(ls -A "$ENCRYPTED_DIR")" ];then
        #Builds an ISO of the .gpg files in the directory unless one is already present
        if [ ! -f $ENCRYPTED_DIR/$ISO_NAME ]; then
                echo "Building ISO to be burned to the disk"
                /usr/bin/mkisofs -o "$ENCRYPTED_DIR/$ISO_NAME" -J -R -A -V -v \
$ENCRYPTED_DIR/
        else
                echo "ISO is already present preparing to burn it to disk"
        fi
else
        echo "There are no logs to be offloaded today the $TODAY."
        echo "They may have already been offloaded please verify with the other \
Administrators."
        exit
fi
#Checks to make sure the ISO is present and if it is this removes the gpg files that were \
placed in it.
```

Continued on next page….

```
#The ISO is then burned to the disk.
if [ -f $ENCRYPTED_DIR/$ISO_NAME ]; then
        echo "Removing files used to build ISO"
        echo ""
        find $ENCRYPTED_DIR -name "*.gpg" -exec rm -rf {} \;
        echo "Burning ISO to disk"
        /usr/bin/cdrecord -v speed=4 dev=$DEVICE $ENCRYPTED_DIR/$ISO_NAME
else
        echo "$ISO_NAME was not created and as such cannot be burned to disk"
        echo "Please troubleshoot the problem or re-run this script"
        exit
fi
#Checking to see if the ISO was successfully burned and if it was the script will email \
the administrators and security personnel.
echo "The Logs have been offloaded emailing all parties notified that the disk needed to \
be burned."
echo "They are being informed that the task is complete."
echo ""
echo ""
echo "Checking for file on disk and then deleting $ISO_NAME in $ENCRYPTED_DIR"
[ -d $MOUNT ] || mkdir -p $MOUNT
/bin/mount $MOUNT_DEV $MOUNT
# Check for mount
grep -qs “$MOUNT” /proc/mounts >> /dev/null
# Check for presence of ISO at the mount
if [ $? -eq 0 ]; then
        # Check to see if file is within the mount
        ll "$MOUNT" |grep -qs "$BURNT_NAME" >> /dev/null
        if [ $? -eq 0 ] ; then
```

Continued on next page….

```
                echo “File was found on the disk after burning”
                echo “Now deleting the resident files left after the burn”
                # Deleting the ISO on the hard disk
                find $ENCRYPTED_DIR -name "*.iso" -exec rm -rf {} \;
                echo "Back-up complete have a great day."
                /bin/mail -s "$SUBJECT" "$EMAIL_IA" < $EMAILMESSAGE
                /bin/mail -s "$SUBJECT" "$EMAIL_ADMIN" < $EMAILMESSAGE
        else
                echo “The file did not burn properly, please contact $EMAIL_ADMIN!”
                /bin/mail -s “$SUBJECT_BAD” “$EMAIL_ADMIN” < $BADMESSAGE
                /bin/mail -s "$SUBJECT_BAD" "$EMAIL_IA" < $BADMESSAGE
        fi
fi
/bin/umount $MOUNT
echo "Ejecting disk"
/usr/bin/eject $MOUNT_DEV
```

Change the ownership to root account:

```
chown root:root /root/backup_mail_logs
```

Adjust permissions for the script to read, write and execute for root:

```
chmod 4700 /root/backup_mail_logs
```

Create the symbolic link to the disk burning script:

```
ln -s /root/backup_mail_logs /data/burn_to_disk
```

Edit the ‘crontab’ for the root user:

```
crontab -u root -e
```

Add the following to the ‘crontab’:

```
# Custom Scripts
59 * * * * /root/log_move
```

Continued on next page….

```
0 23 * * * /root/log_archiver
30 23 * * 7 /root/move_compressed_logs
50 23 * * * /root/last_day_of_month && /root/end_of_month_consolidation
0 1 1 * * /root/encrypt_move_mail_logs
```

Appendix B: Suggested Readings for Hardening the Audit Server and the GWOS Server

Burgiss, H. (July 2, 2002) Security Quick-Start HOWTO for Linux: What services do we really need?, Retrieved April 4, 2012 from http://www.tldp.org/HOWTO/Security-Quickstart-HOWTO/services.html

CAM Table Overflow (n.d.) in Hakipedia. Retrieved June 21, 2012 from http://hakipedia.com/index.php/CAM_Table_Overflow

CentOS 6 Minimal or Hardened Install (January 19, 2012) In Admin's Goodies To be a good sysadmin! Retrieved January 24, 2012 from http://admingoodies.com/centos-6-minimal-or-hardened-install

Gerards, R. (July 22, 2005) SSL Encrypting Syslog with Stunnel. Retrieved April 20, 2012 from http://www.rsyslog.com/doc/rsyslog_stunnel.html

Gerards, R. (May 6, 2008) Encrypting Syslog Traffic with TLS (SSL). Retrieved April 20, 2012 from http://www.rsyslog.com/doc/rsyslog_tls.html

Gerhards, R. (March, 2009). The Syslog Protocol. Internet Engineering Task Force (IETF) Request for Comments (RFC) 5424. Retrieved from http://tools.ietf.org/html/rfc5424

Hayden, M. (January 1, 2012). Getting started with SELinux. Retrieved from http://rackerhacker.com/2012/01/25/getting-started-with-selinux/

Maro, T. (March 10, 2009) Stop Port Scans In Their Tracks With IPTables, Retrieved January 25, 2012 from http://www.ossramblings.com/using_iptables_rate_limiting_to_prevent_portscans

Miao, F., Ma, Y., & Salowey, J. (March, 2009) Transport Layer Security (TLS) Transport Mapping for Syslog. Internet Engineering Task Force (IETF) Request for Comments (RFC) 5425. Retrieved from http://tools.ietf.org/html/rfc5425

Nataranjan, R. (June 14, 2011) 25 Most Frequently Used Linux IPTables Rules Examples, Retrieved January 30, 2012 from http://www.thegeekstuff.com/2011/06/iptables-rules-examples

New, D., & Rose, M. (November, 2001). Reliable Delivery for syslog. Internet Engineering Task Force (IETF) Request for Comments (RFC) 3195. Retrieved from http://www.ietf.org/rfc/rfc3195.txt

Vivek Gite (October 19, 2005) How Do I Secure Grub Boot Loader? Retrieved January 31, 2012 from http://www.cyberciti.biz/tips/how-do-i-secure-grub-boot-loader.html

Walsh, D. (August 21, 2007) A step-by-step guide to building a new SELinux policy module. Retrieved from http://magazine.redhat.com/2007/08/21/a-step-by-step-guide-to-building-a-new-selinux-policy-module/

Zang, H (n.d.) Three attacks in SSL protocol and their solutions. Retrieved from http://www.cs.auckland.ac.nz/courses/compsci725s2c/archive/termpapers/725zhang.pdf

Last Updated: July 20th, 2018

Upcoming SANS Training

Click here to view a list of all SANS Courses

SANS Riyadh July 2018	Riyadh, SA	Jul 28, 2018 - Aug 02, 2018	Live Event
SANS Pittsburgh 2018	Pittsburgh, PAUS	Jul 30, 2018 - Aug 04, 2018	Live Event
Security Operations Summit & Training 2018	New Orleans, LAUS	Jul 30, 2018 - Aug 06, 2018	Live Event
SANS Hyderabad 2018	Hyderabad, IN	Aug 06, 2018 - Aug 11, 2018	Live Event
Security Awareness Summit & Training 2018	Charleston, SCUS	Aug 06, 2018 - Aug 15, 2018	Live Event
SANS Boston Summer 2018	Boston, MAUS	Aug 06, 2018 - Aug 11, 2018	Live Event
SANS San Antonio 2018	San Antonio, TXUS	Aug 06, 2018 - Aug 11, 2018	Live Event
SANS August Sydney 2018	Sydney, AU	Aug 06, 2018 - Aug 25, 2018	Live Event
SANS New York City Summer 2018	New York City, NYUS	Aug 13, 2018 - Aug 18, 2018	Live Event
SANS Northern Virginia- Alexandria 2018	Alexandria, VAUS	Aug 13, 2018 - Aug 18, 2018	Live Event
SANS Krakow 2018	Krakow, PL	Aug 20, 2018 - Aug 25, 2018	Live Event
Data Breach Summit & Training 2018	New York City, NYUS	Aug 20, 2018 - Aug 27, 2018	Live Event
SANS Chicago 2018	Chicago, ILUS	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Prague 2018	Prague, CZ	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Virginia Beach 2018	Virginia Beach, VAUS	Aug 20, 2018 - Aug 31, 2018	Live Event
SANS San Francisco Summer 2018	San Francisco, CAUS	Aug 26, 2018 - Aug 31, 2018	Live Event
SANS Copenhagen August 2018	Copenhagen, DK	Aug 27, 2018 - Sep 01, 2018	Live Event
SANS SEC504 @ Bangalore 2018	Bangalore, IN	Aug 27, 2018 - Sep 01, 2018	Live Event
SANS Wellington 2018	Wellington, NZ	Sep 03, 2018 - Sep 08, 2018	Live Event
SANS Amsterdam September 2018	Amsterdam, NL	Sep 03, 2018 - Sep 08, 2018	Live Event
SANS Tokyo Autumn 2018	Tokyo, JP	Sep 03, 2018 - Sep 15, 2018	Live Event
SANS Tampa-Clearwater 2018	Tampa, FLUS	Sep 04, 2018 - Sep 09, 2018	Live Event
SANS MGT516 Beta One 2018	Arlington, VAUS	Sep 04, 2018 - Sep 08, 2018	Live Event
Threat Hunting & Incident Response Summit & Training 2018	New Orleans, LAUS	Sep 06, 2018 - Sep 13, 2018	Live Event
SANS Baltimore Fall 2018	Baltimore, MDUS	Sep 08, 2018 - Sep 15, 2018	Live Event
SANS Alaska Summit & Training 2018	Anchorage, AKUS	Sep 10, 2018 - Sep 15, 2018	Live Event
SANS Munich September 2018	Munich, DE	Sep 16, 2018 - Sep 22, 2018	Live Event
SANS London September 2018	London, GB	Sep 17, 2018 - Sep 22, 2018	Live Event
SANS Network Security 2018	Las Vegas, NVUS	Sep 23, 2018 - Sep 30, 2018	Live Event
SANS DFIR Prague Summit & Training 2018	Prague, CZ	Oct 01, 2018 - Oct 07, 2018	Live Event
Oil & Gas Cybersecurity Summit & Training 2018	Houston, TXUS	Oct 01, 2018 - Oct 06, 2018	Live Event
SANS Brussels October 2018	Brussels, BE	Oct 08, 2018 - Oct 13, 2018	Live Event
SANS Pen Test Berlin 2018	OnlineDE	Jul 23, 2018 - Jul 28, 2018	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced