

FOSS Log Management that Works!

MAKING SENSE OF THE MADNESS WITH DAD

 Cyber-Defense

FOSS Log Management

Authored by David Hoelzer

The contents of this presentation are copyright ©2007, all rights reserved by Enclave Forensics and CyberDefense. No part of this book may be reproduced without express written consent from an authorized representative of CyberDefense or its parent organization, Enclave Forensics.

Cyber-Defense

CyberDefense is the open source development arm of Enclave Forensics, a member of the Enclave family of companies. By using this book and the accompanying software you agree to hold harmless CyberDefense, Enclave Forensics and their respective officers for any and all damage that might occur, physical or otherwise, through the use and application of the techniques and software described here. The software presented that is copyrighted by CyberDefense is free for your use. We do request, however, that you direct interested individuals to the SourceForge repository to obtain their own copy of the software rather than redistributing it directly.

Table of Contents

FOSS Log Management	2
Introduction	6
DAD Basics.....	13
Why WAMP?	13
Domain Integration.....	14
Major DAD Components	15
Role Based Access Controls	16
Extensibility	17
Localization	17
Behind the Scenes.....	18
Priority System	19
Dynamic Scaling	20
Scheduling in Java	23
Alerting.....	23
Syslog	26
Log Carver	27
DAD Interface.....	30
Installing DAD.....	33
OS.....	35
Unpacking DAD	35
Installing DAD.....	36
MySQL.....	36
Apache	36
PHP	36
Perl	37

EXTREMELY IMPORTANT:	37
Post Installation	38
Troubleshooting.....	38
Configuring DAD.....	40
MySQL.....	40
Apache	41
PHP.....	41
Final Installation.....	43
Domain Configuration.....	45
SDDL Explained	47
Adding DACLS to Group Policy	48
Adding Servers and Computers to Process	50
Regular Expressions	52
[A-Z][a-z]{2}.*sshd\[[0-9]+\]	55
.*localhost.*for user	55
.*Failed.*	55
Alerting.....	58
Scalability	60

Roadmap

- Log Management Problems
- DAD – What it is and How it Works
- Installing DAD, Step by Step
- Using and Configuring DAD



The goal in this short course is to provide you with a tool that can solve the majority of the log management and alerting problems that businesses face today. We will begin by examining some of the common problems that we face in log management in addition to discussing some of the obstacles to managing logs effectively. Following this, we will look at a possible solution to most log management problems through a free and open source software (FOSS) tool called DAD.

To round out the course, you will have the opportunity to install DAD and then to spend the majority of the course configuring, using and learning about the tool itself. This should equip you to return to your office and begin using DAD immediately if you so choose.

Problems in Log Management

- Quotes I've heard personally:
 - "It doesn't really do what the vendor said."
 - "It works fine until you start putting events into it."
 - "Every time we patch, all of the agents break."
- Do you have a system that you like?



Introduction

Over the last eight years, I've spent a significant amount of time training auditors and administrators how to secure, manage and monitor their systems. One of the things that have always interested me is that every class knows that system administrators should be examining their logs every day. This, in fact, is best practice. Unfortunately, the reality is that most administrators are not looking at their logs every day. In fact, most administrators are only looking at system logs when there's something wrong and they are trying to trouble shoot the problem. This is probably the worst possible time to look at your logs since, at this point, you really don't know what they look like normally!

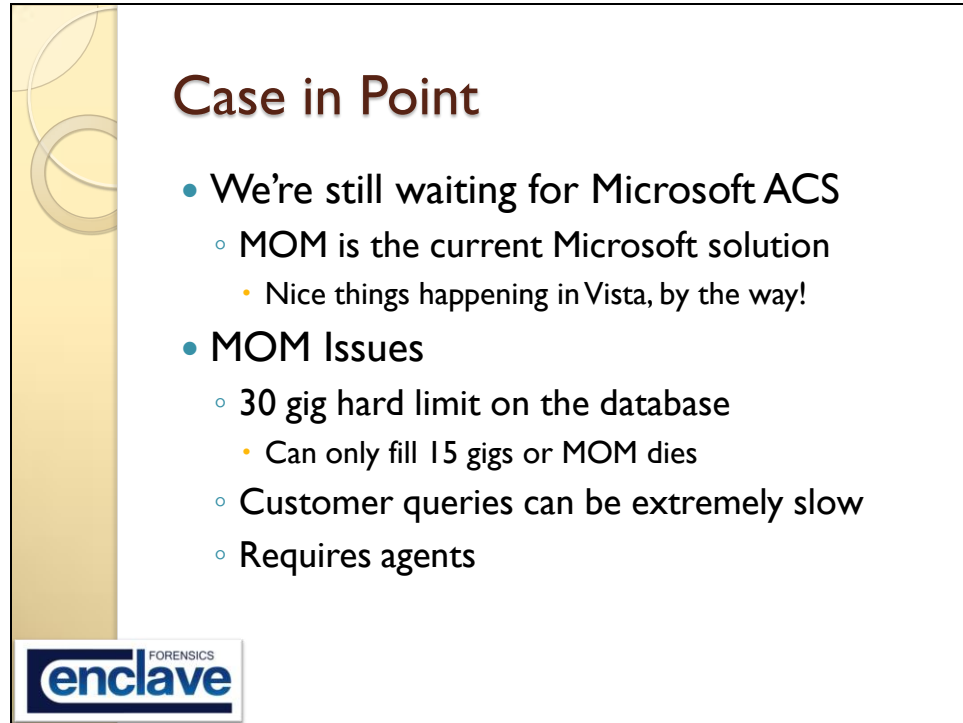
As a result of this harsh reality, and sometimes to satisfy regulatory or contractual obligations, many enterprises are either implementing or have implemented some sort of enterprise log management system. Here too there are some unfortunate realities.

At the time of this writing, we have polled literally hundreds of administrators and auditors for feedback regarding log management tools. While most are managing to get by with the log management system that they have in place, almost all report that they are really not satisfied with the log management tool that they have, regardless of the cost.

Aside from complaints about how the tool works in reality compared with how the vendor claims that the tool functions, the number one complaint has to do with the requirement for an agent on local systems. System administrators are extremely reluctant to install agents on the systems that they manage. This reluctance is well founded.


The installation of an agent tends to introduce an unknown into what may already be a somewhat unstable system. Worse, if the system is currently stable, the administrators have likely worked long and hard to work out all of the bugs to keep the system stable. The introduction of an agent creates the potential for yet another source of instability.

As an illustration of the seriousness of this problem please consider Microsoft Operations Manager. Several sites where I have seen this installed report that “Update Tuesday” is a day of dread for the MOM administrators. Apparently, even though MOM is a Microsoft product, the patches released through Microsoft will quite often cause critical failures in the MOM infrastructure, particularly among the MOM agents.

A presentation slide titled "Case in Point" with a yellow and white background. It contains a bulleted list of points about Microsoft ACS and MOM. The Enclave Forensics logo is in the bottom left corner of the slide.

Case in Point

- We're still waiting for Microsoft ACS
 - MOM is the current Microsoft solution
 - Nice things happening in Vista, by the way!
- MOM Issues
 - 30 gig hard limit on the database
 - Can only fill 15 gigs or MOM dies
 - Customer queries can be extremely slow
 - Requires agents



Another one of the serious issues that I find administrators and auditors are contending with when it comes to log management tools are the limits that these tools impose. Microsoft Operations Manager is also an excellent example in this area.

Microsoft has long promised (for at least six years now) the release of "Audit Collection Services." This tool is supposed to be the next generation for log aggregation and management in a Windows domain. Unfortunately, so much time has passed that it simply does not seem realistic to rely on vaporware of this type to solve our problems.

This is not to say that there hasn't been any progress. If you've had the opportunity to look at Microsoft Vista, you know that there have been some significant advances in the Event Viewer. The new Event Viewer allows you to subscribe to multiple event sources. You are also able to handle logs as XML rather than the proprietary Windows event log format. Even so, this is not really an enterprise solution to centralized log management. For this, MOM is still the Microsoft solution.


To be honest, we have not installed the latest version of MOM. With that disclaimer, the versions of MOM that we have worked with have some very real limits. For instance, MOM has a hard limit of 30 gigabytes for the size of the MOM database. This doesn't sound like a tremendous limit, but

in fact you can only populate half of that size. Once the database grows beyond 15 gigabytes, MOM no longer has enough free space for temporary tables, sorting and grooming. What this means is that once you pass 15 gigabytes, a DBA must act to reduce the size of the tables manually or MOM will eventually die.

You may think, “Still, 15 gigabytes is a lot of logs!” For some this may be true. One site that we are monitoring, however, generates more than 8 gigabytes of event logs per day! This limit has forced the site in question to monitor only a few servers and to limit which events they collect to prevent MOM from melting down.


Another issue to contend with in MOM and most other log management solutions is that the built in queries tend to work well, but customized queries can be quite slow. We’ll likely mention a few examples for comparison later on today.

The last major problem is that, like most tools, MOM requires agents in order for it to work correctly. Many commercial solutions advertise that they have an agentless collection system, but many administrators have reported back that in order to use some products to the best effect an agent must be installed on the monitored clients.



Other Issues

- How tightly does the tool integrate with the rest of my environment?
 - Can I handle Syslog?
 - Many sources
 - Routers, firewalls, IDS, etc.
 - What about text based log files?
 - Custom application logs, etc.



The other feature that many organizations are looking for in a log management tool is the ability to handle syslog messages from routers, firewalls, UNIX systems and more. As it turns out, accepting syslog is actually very simple. All that is really necessary is a UDP listener on port 514. The thing that is more difficult is turning the syslog messages into something that is actually usable.

The trouble with syslog messages is that aside from the timestamp and reporting system, the remainder of the message can take almost any form. This problem is not limited to syslog, however. It is common for enterprises of any size to have internally developed applications, most of which generate some type of log information. Any full featured log solution should also be able to import these arbitrary logs into a usable format.

What if I told you...

- How about something that does this:
 - Near real time event aggregation
 - No hard limit on data sources
 - No hard limit on data storage
 - Handles Windows event logs (without agents)
 - Digests Syslog
 - Digests any text based log (with “training”)
 - Can be scaled up to thousands of hosts
- What if we told you it was free*?




If the issues outlined so far sound like issues that you have dealt with, what would you say if we told you that we had a tool that can provide the following:

- Near real time
- Windows event log aggregation
- Syslog messages
- Arbitrary text based logs
- No hard limit on the number of events stored
- Dynamic alerting
- Fast searching
- Automatic grooming

What would you say if we told you that you can have such a tool for no charge? Of course, you may not believe that such a tool exists just yet, but you can judge for yourself at the end of the presentation. Also, it is important to realize that while there is no cost to the actual software used to aggregate and manage the logs, there is certainly a cost associated with the procurement of the hardware to run such


a system. Even so, a fairly modest system can handle a great deal of information with the system that we are proposing for you.

As an example, one of the current development servers that we are using for internal development and testing is a dual Pentium III running at 500 megahertz and only 1 gigabyte of RAM. Even so, that system is doing quite well at aggregating and alerting on events in a relatively small test domain. Of course, we do not recommend this configuration for a live deployment of this tool. We will discuss considerations along these lines later in the course today.



What DAD is

- **WAMP Based**
 - Windows, Apache, MySQL, PHP (Mostly...)
 - Allows for easy integration into a Windows domain by a Windows administrator
 - Requires only one right in the domain
 - Manage and Audit Security Logs
 - Also requires an ACL for other logs



DAD Basics

Let's get down to business and explain to you exactly what it is that we're talking about here. DAD (Distributed Aggregation for Data analysis) is a WAMP(Windows/Apache/MySQL/PHP)based solution built entirely on free and open source software. Many people have asked, "Why is it WAMP based rather than LAMP based?" There are two very simple answers to this question.

Why WAMP?

In creating this tool we were trying to solve one very serious problem: how to manage Windows based logs. Generally speaking, UNIX administrators have long had the capability to aggregate their logs using built in portions of the operating system. UNIX administrators also have at their disposal a number of free scripts that can be very useful in processing text based logs, which is what UNIX logs are for the most part. Tools of this sort include Swatch (<http://sourceforge.net/projects/swatch/>) and SEC (<http://sourceforge.net/projects/simple-evcorr/>). These tools, and others like them, provide the ability to search and alert on events or even to provide triggered alerting based on multiple events.

While this is fine for UNIX administrators, Windows administrators have long wrestled with the issue of how to effectively gather all of the events into one location and to then process them

effectively. This is exacerbated by the fact that Windows events tend to be spread all over the domain rather than centrally stored on the domain controllers.

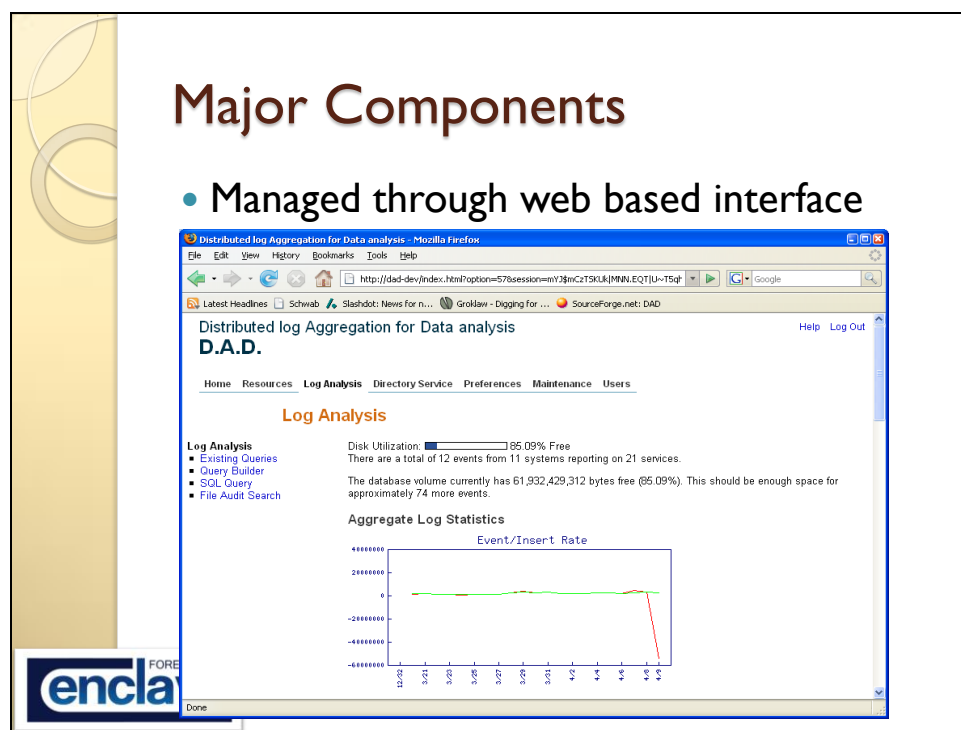
Since the tool was initially created to solve these major problems for Windows administrators, a conscious decision was made to create a Windows based solution. It was felt that this would make the entire solution much more acceptable within a Windows environment since there can be strong opinions when it comes to the introduction of a UNIX based solution to manage Windows hosts.

A second reason why we have remained on a Windows platform is twofold. While it is possible to configure Windows Kerberos to work with a UNIX based Kerberos system, it is actually necessary to make the Windows Kerberos communication less secure to do so. Additionally, there is a very convenient Perl module that is capable of reading event logs remotely using native Windows RPC communication. This module is only available on Windows. 😊

Domain Integration

We will go into more detail later in the next section, but one of the other items that were very important for the developers was ensuring that the system could run within a Windows domain with the minimum privileges required. There are a number of solutions available today that require Domain Administrator credentials to operate. Others require that the agents operate under the Local System credential or with Domain Administrator credentials. These models are very disturbing when considered against the best practice principle that the minimum privilege required for a task be used.

For DAD to function correctly, we recommend that you create a service account for the scheduler to run under. This account requires only the right to manage and audit security logs within the domain. We will demonstrate how to configure this with group policy later today.



Major DAD Components

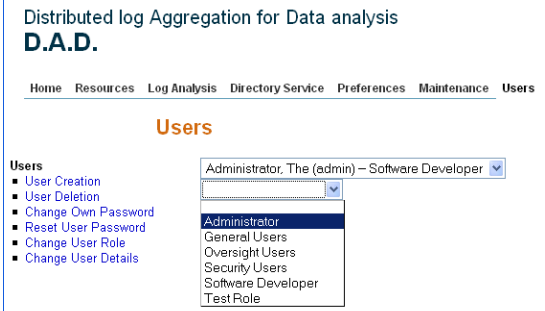
Let's take a few minutes to go over the major components that make up DAD. First of all DAD is managed through a PHP based web interface. All major functions can be controlled directly through this interface with no need to poke around under the hood unless you really want to.

Future versions of DAD may include more functions to allow you to interface with the Active Directory or another LDAP server or possibly to monitor other types of resources. There are also future plans of incorporating basic health and resource monitoring for systems and the network in general. We even have a few items in place in the schema to perhaps expand out for full network discovery and monitoring from within DAD. These features are at least a year away, however.

We are moving toward the notion of a "dashboard" within DAD, as you can see in the slide pictured above. As time goes on, you will see the dashboard continue to become better defined as reporting modules are completed and integrated with the scheduler. It is likely that the dashboard that you see at the conference where you are attending this course will look somewhat different than the one pictured above since we are constantly working to improve the usability and usefulness of the interface.

Interface Components

- Interface is almost completely PHP based
 - Driven by SQL backend
 - Modular design for extensibility
 - Strong sense of ACLs implemented through Roles



Another feature that is rather important when it comes to security monitoring and alerting is whether or not the tool can be widely deployed. What we mean is, "can I give the system administrators access to the system so that they can manage the logs themselves?" The answer is, again, yes!

Role Based Access Controls

Within DAD, every single menu item that is displayed through the interface is controlled through a role based access control system. This means that you can assign exactly which roles should be able to see and execute each menu option. In fact, this concept is driven all the way through the product. It is also possible to assign roles to the various queries that can be run through the log management interface.

In terms of security this allows us to carefully control exactly which queries any individual might be able to run. For instance, there may be data stored within some of the events that contains sensitive information. It may not be appropriate to permit every DAD user to see every log entry.

In an effort to force administrators to use role based access control, it is impossible to assign rights to a menu or query to a specific user. The only way that rights can be assigned is by role. It is also

important to keep in mind that some of the menu items are quite powerful and can even be used to bypass the security systems within DAD. For instance, it is possible to enter raw SQL queries through the log management interface. Clearly, only administrators should have this ability.

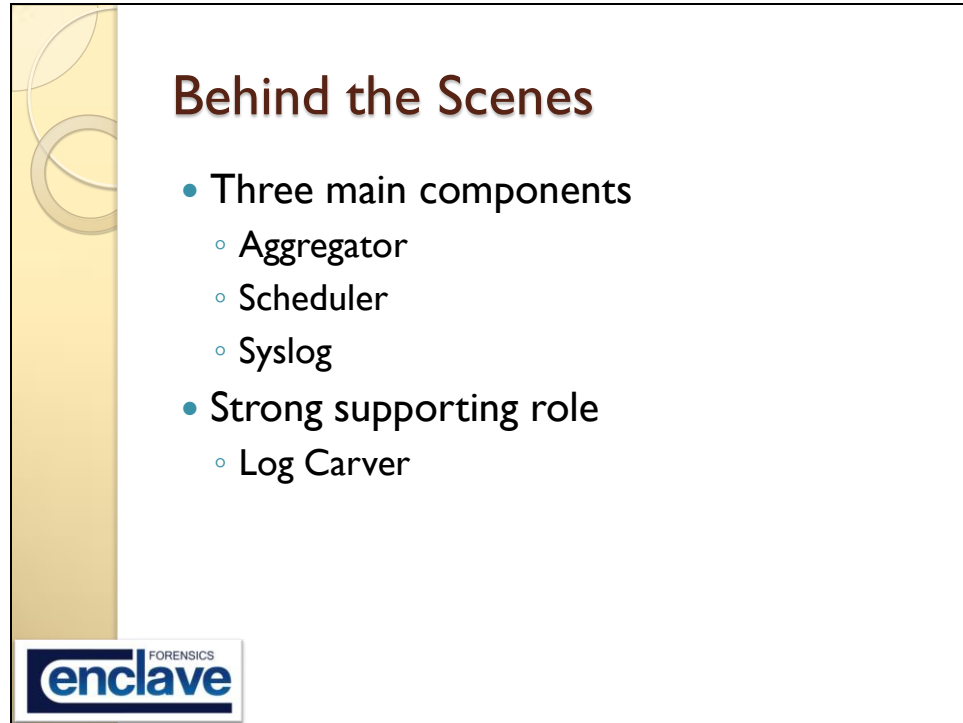
Extensibility

We have put a great deal of effort into making the system as open and extensible as possible. This means that with a little bit of effort spent learning how DAD functions internally, you should be able to unleash your own internal programmers on DAD to add new menu options and other functionality with a minimum of effort.

Even so, we readily admit that the current menu system within DAD, while functional, is especially esoteric. There are functions built into the web interface to make the addition of menu options a bit easier, but it still could be better. We do have plans to revisit the menu management pieces of DAD in the coming months. Since the menus are currently functioning well, we feel that it would be better to spend time adding functionality at the moment. If you have time or resources to devote to cleaning up the menus, please feel free to let us know!


Localization

Since we believe in building for the future, DAD also incorporates a localization system so that over time we can provide native language support for many languages besides English. While there has been some effort put forward to ensure that all of the DAD menus can be easily localized, the reporting modules and the SQL queries do not currently support localization. Sometime before September 1, 2007, the DAD team plans to fully normalize the events table to speed queries and optimize the number of events that can be stored. After this work is completed, the team plans to standardize the localization and provide an interface within DAD itself to work with the localization information more easily. At the same time, localization will be enforced throughout DAD.

A presentation slide titled "Behind the Scenes" with a yellow vertical bar on the left containing a circular graphic. The slide lists three main components and a supporting role. The Enclave Forensics logo is in the bottom left corner of the slide.

Behind the Scenes


- Three main components
 - Aggregator
 - Scheduler
 - Syslog
- Strong supporting role
 - Log Carver



Behind the Scenes


There are really only three main pieces to DAD once you get past the web interface. Of those, one acts as the master service that starts all other processes. These three processes are the aggregator, the scheduler and the syslog daemon.

The scheduler is really the most important task on the DAD server. The scheduler is a Java service that runs in the background watching and monitoring all of the other DAD processes that need attention. The scheduler is managed through the web interface as are the other pieces of DAD. Even so, the aggregator gets the most attention since it does the majority of the work, so we'll start our focus there.



Aggregator

- Currently a Multithreaded Perl Script
 - Dynamically polls events out of Windows systems
 - Priority system for polled systems
 - Schedules polling dynamically based on event rate
 - Runs with reduced credentials (not Domain Admin)
 - Parses event data and inserts to SQL backend
 - Several aggregate fields are created for faster searching



The DAD Aggregator is a multithreaded Perl script that polls all configured systems for log changes periodically. How often the systems are polled and what criteria is used for scheduling are important concerns that we will deal with here. If we put those issues aside for just a moment, then at its simplest the aggregator pulls logs from remote Windows systems, parses those logs into various fields and inserts the events into the back end database.

Priority System

Within your enterprise, there are some systems whose logs you are more interested in, or whose logs simply roll over faster than others. For this reason, DAD has the capability to dynamically scale how frequently it will request log changes from any particular server. The main driving control for this is the priority of the systems in question.

DAD currently does not limit the number of priority levels that you might choose to use. Even so, we would encourage you to exercise good sense. It is likely that you could actually get DAD to do something unpredictable if you were to configure it with a negative priority or with an exceedingly large priority. At the moment, we have not taken steps to prevent this since the product is still in an Alpha

stage and only an administrator should be adding systems to monitor anyway. The priorities recommended typically used range from zero through ten.

Priority zero is currently used to indicate that the monitoring of the given system is disabled. This is useful during the time period between the time that a system is decommissioned and the time that all events generated by that system have been groomed out of DAD. If the system were simply removed, DAD would have no way to map these events back to a particular system. This feature is also useful in debunking DAD as the cause of crashes, particularly on NT 4 systems. Simply disable the log polling and see if the NT 4 system recovers.

Priority one is specifically for systems that need to be monitored in real time and which should take precedence over all other systems being monitored. Priority one systems will always be polled frequently even if they have very few or no events available. Priority one systems are also given the greatest slice of polling time. Typically domain controllers and other critical servers would be configured as priority one systems.

Priority two systems are allotted less total time for polling and, depending on how many events they are generating, may be polled less frequently than priority one system. In fact, all of the priorities below two continue to degrade the frequency of polling and the maximum polling time. Priority two systems are typically critical servers like print servers and file servers. Operations management workstations might also be included here.

Systems scheduled with a priority greater than two would typically begin to represent desktops. Systems that are frequently offline should probably be configured at priority four or five.

Dynamic Scaling

As you can see, the priority system has an influence over how long a given system might be polled during any particular cycle. The other issue that the priority affects has been mentioned as well; that is the amount of time that will pass between polling cycles.

The aggregator itself pulls the list of systems being monitored from the database every few seconds. As these systems are polled, the aggregator keeps track of how many new events are found on each system when they are polled. Using the number of events discovered and the priority of the system the aggregator calculates when the next poll should occur.

When this calculation is made, the aggregator degrades the polling time progressively. What this means is that if a system being polled has no new events, the aggregator will not immediately schedule the next poll for that system to be the maximum delay. Instead the aggregator will progressively increase the amount of time for each system based on the number of events collected. Each time that the system is polled and the number of events polled is low or zero, the aggregator will increase the polling delay until a reasonable delay is discovered or a maximum threshold is reached. The thresholds are determined based on the priority of the system being monitored.

Aggregator in Action

- Future is to rewrite Event log component and re-implement in Java for portability


```

Command Prompt - Aggregator.pl
2 Missed 176327 events from XEONDC-2G!! Catching up.
1 Missed 4809251 events from AGAMEMNON!! Catching up.
0 Missed 1 events from ZEUS!! Catching up.
0 Missed 1 events from ZEUS!! Catching up.
5 Missed 1 events from DAD-DEU!! Catching up.
7 Missed 1 events from DEMETER!! Catching up.
7 Missed 1 events from DEMETER!! Catching up.
4 Missed 1 events from HERA!! Catching up.
2 Missed 1 events from XEONDC-2G!! Catching up.
1 Missed 1 events from AGAMEMNON!! Catching up.
0 Missed 1 events from ZEUS!! Catching up.
0 Missed 1 events from ZEUS!! Catching up.
7 Missed 1 events from DEMETER!! Catching up.
7 Missed 1 events from DEMETER!! Catching up.
4 Missed 1 events from HERA!! Catching up.
4 Missed 1 events from HERA!! Catching up.
Likely log reset on ZEUS. DNS Server log. Forcing pull.
1 Missed 2315 events from AGAMEMNON!! Catching up.
Likely log reset on DEMETER. DNS Server log. Forcing pull.
3 Missed 2014 events from TABLET!! Catching up.
3 Missed 2311918 events from TABLET!! Catching up.
3 Missed 1 events from TABLET!! Catching up.

```

Even though the aggregator works well as it is, there is some serious discussion that may result in the migration of the aggregator code to Java. This would allow for even greater portability of the code and would eliminate the need for external Perl modules. The downside is, of course, that Java does not have the same facility with string manipulation that Perl has.

Even if the code is not migrated to Java, there is real interested in re-implementing the piece of supporting code that is currently used to extract event logs from Windows hosts. This code is not very actively maintained and requires a Windows environment to function. This means that while all of the other parts of DAD can be migrated to any platform, the aggregator itself *must* be run on a Windows platform at present.



Scheduler

- Allows for the creation of custom jobs and/or reports
 - Allows for population of the “dashboard”
 - Allows for alerting
 - Cares for maintenance (grooming, stats, etc.)

Jobs

Jobs: <input type="button" value="Update"/> <input type="button" value="Save as New"/> <input type="button" value="Delete"/> <input type="button" value="New"/> <input type="button" value="Refresh"/>		
Description:	Start Time: 08:34	Job ID:
Job Type:	Start Date: 2007-04-09	Times Ran:
Path to Script:	Username:	Last Ran:
Package Name:	DN:	Caller Active:
Times to Run:	Password:	Time Active:
Hour/Minute: * <input type="text"/> *	Day of Week: * <input type="text"/> *	
Day of Month: * <input type="text"/> *	Month of Year: * <input type="text"/> *	

The scheduler is the heart and soul of DAD. Using the web based interface, it is possible to configure jobs to run at almost any frequency that you can imagine. These scheduled jobs can be in any language or executable format that is supported on the platform that the scheduler is running on. If you happen to use Java, you can even leverage the scheduler itself.

Scheduling in Java

The scheduler, like the rest of DAD, is open source. If you choose to develop scheduled jobs using Java, it is possible to signal other jobs through the DAD scheduler itself. The primary and likely most important part of this framework allows one job to send another job a message. This can, of course, go in both directions, essentially creating a primitive inter process communication architecture for scheduled jobs.

Alerting

The main function of the scheduler, aside from making sure that the aggregator is always running, statistics are periodically regenerated and that the groomer is cleaning out old events from time to time, is to allow DAD to perform some automatic correlation and alerting. Essentially, if there's something that you would like to alert on, you simply need to create a job and schedule it!

Included with the current release of DAD there are a few interesting alerts that can be scheduled. We have also included the basic code required to create an alert that can populate the dashboard in Java, Perl and PHP. These samples can easily be modified to perform any sort of regular analysis that you might be interested in.

Scheduler Details

- Multithreaded Java backend
 - Jobs managed through the database
 - Job definition can execute any external code
 - Perl, Java, executable, etc.
 - Runs as a Windows service
 - Designed to run with low credentials
 - Considering implementing the ability to execute jobs with alternate credentials but we have concerns about the security implications
 - Storage of credentials, authentication, etc.



The jobs themselves are managed through the web interface and stored in the SQL back end. The scheduler periodically polls the database to determine which jobs should have started or should have ended and then makes sure that appropriate actions are taken or that an alert is generated to indicate that there is a problem.

The scheduler runs as a Windows service. Currently, we recommend that the scheduler run with the same credentials that DAD polls your Windows event logs with. This should be an account with very low privileges, not a Domain Administrator. There has been some discussion about whether or not to allow the scheduler to start jobs with alternate credentials and this is currently reflected in the job scheduling interface.

The current release version of DAD will not honor alternate credentials for jobs. It is uncertain whether or not this feature will be completed or removed since there are some serious security concerns when it comes to storing credentials in the DAD tables.

Syslog Listener

- Collects syslog events into a local log file
 - Scheduler periodically (default is every two minutes) kicks off the Log Carver
- There's a lot of interest in going the other way
 - In other words, something to push Windows events to Syslog without agents
 - We've included a modified version of the aggregator in DAD that can do just this




Syslog

Besides the aggregator, the other main job that the scheduler always keeps running is the syslog listener. The syslog listener simply grabs all syslog messages that are delivered to UDP port 514 and relays them into a text file for later processing.


Since syslog messages are so varied in form it was decided that it made very little sense to try to implement syslog parsing within the syslog listener itself. Instead, the existing code in the log carver was leveraged to massage the data into a usable format and subsequently insert that data into the database. We'll discuss the log carver shortly.

There has been a great deal of interest from UNIX administrators in joining the functionality of the aggregator and the syslog listener. Most of the solutions that allow you to gather your Windows events into UNIX require that you install an agent or a syslog service on your Windows hosts. Included in the current release of DAD we also provide a Perl based solution that will aggregate all of your Windows events and relay them to your syslog servers. This is not actually a part of DAD and will likely be branched off of the DAD code in the near future.



Log Carver

- Capable of carving up anything
 - Minor assembly required 😊
 - Comes capable of carving up:
 - Various Syslog messages:
 - Postfix
 - Sendmail
 - BIND
 - IPTables
 - Can be run against any kind of log




Log Carver

In the last few pages we mentioned that the syslog service actually does not deliver data into the database but instead stores the data into a file. Every few minutes the scheduler asks the syslog service to roll this file over and then starts the log carver. The log carver does the real legwork.


The log carver is currently a Perl script that can be configured to extract any type of interesting log information from any type of text based log file. The data extracted can be parsed into fields and then pushed into the SQL database.

The advantage to this is that the arbitrary carving of logs has been centralized into a single piece of manageable code rather than sprinkled throughout the various pieces of DAD that might require such a facility. The other advantage is that this system does not require that the logs be delivered over the network. To the contrary, the logs can come from any source, even standard input. This also means that it is left within the user's power to teach the log carver how to interpret any kind of log message that it might come across. This does require some minor effort, however.



Carver Configuration

- Requires some knowledge of regular expressions
 - Essentially, you have to teach it how to parse the data into database fields
 - Carving rules are stored in the database
 - Carver modeling performed through web interface
 - Includes the Perl YAPE-Regex engine
 - Allows for testing sample logs to verify output



The minor effort that is required is some knowledge of regular expressions. What we're really talking about is teaching the carver how to recognize where the important data in the log is and how to turn that data into something that can fit into the DAD schema.

The rules that define how data will be carved up are stored in the database as is virtually everything else in DAD. The rules can also be managed through the DAD web interface. In fact, the web interface includes several other features to make this management and modeling easier.

The web interface leverages a small piece of Perl code on the DAD server that runs all expression that the user creates through the YAPE-Regex engine. This engine is most easily understood as a regular expression explanation engine. This allows the user to make sure that the expression that is being created matches the intent.

An additional aid is the ability to paste in sample lines from a log file to examine how the carver will treat them when run against an actual log. This allows the user to figure out which rules will match which lines and how those lines in the log will be digested by the carver before unleashing it on live log files.

What This Means

- You can carve anything
- Example in one enterprise:
 - Custom web application
 - Application logs stored locally on the web server
 - Server at co-location site
 - Nightly, Python job sucks these logs to the internal network with SSH
 - Carver runs automatically to push logs into DAD



What this model also means is that you can carve up absolutely anything. As an example, let's look at one enterprise that has implemented DAD makes use of the log carving engine.

This enterprise has deployed an internally developed Internet facing web application. In order to provide better security for the protected computers in the enterprise, they have co-located their web server at an external site. The web application creates custom logs in a text based format locally on the web server.

Every night, a Python job within the organization reaches out to the web server using SSH with public/private key authentication and pulls back the most current logs stored on the server. This occurs at 7:30 am each day.

Every morning at 8:00 am, the DAD scheduler starts the log carver and points it at the logs that were retrieved previously. The carver now digests the log and parses the useful data into fields that are then inserted into the events database within DAD. At this point, the events are ready for searching or alerting!

What Now?

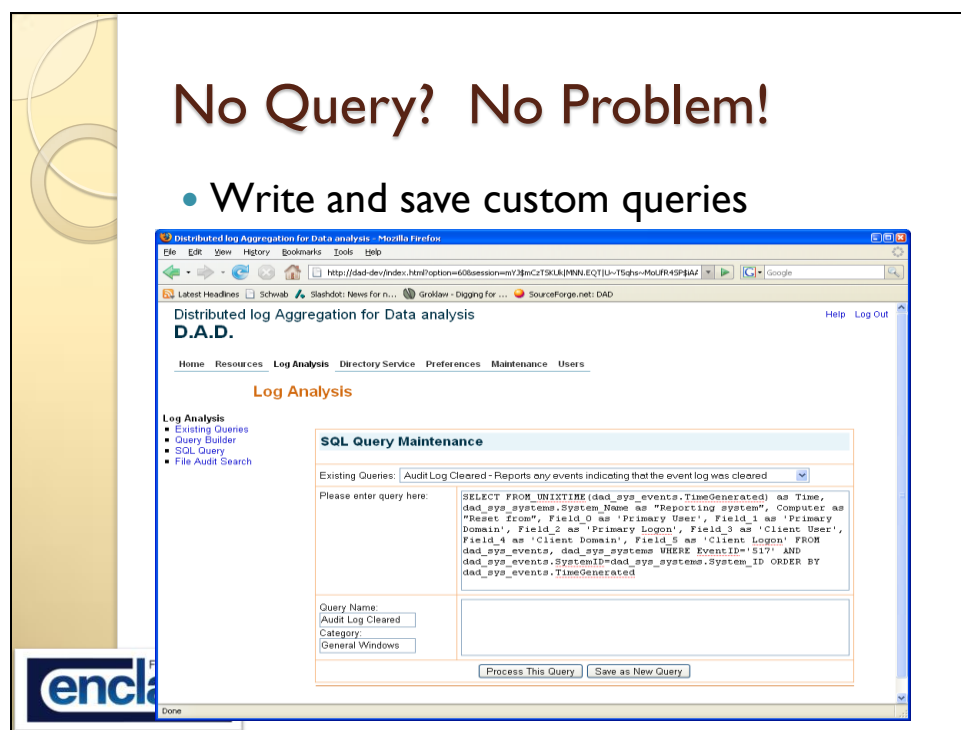
- Once the data is in the database, run queries!

DAD			
Event Count	Event Count by System	Event Count Sorted	Force Repolling
Next Polling Time	Show Services	Windows Event Log Polling	
General Windows			
Account Created	All Logon Failures	Audit Log Cleared	Correlated Logon/Logoff
Correlated Logon/Logoff 24	Domain Joins	Domain Trust Established	Domain Trust Removed
Errors	Errors 24	Failed Interactive	Failed Interactive 24
Failed Network Logons	Failed Network Logons 24	Failed Unlock	Failed unlock 24
Interesting Files	Interesting Files 24	Logon Type Failed	Monitored Resources
NTP Events 60	Printed	Printed 24	Remote Desktop Connections
Updates	Updates 24		
Kerberos			
Account Disabled/Unavailable	Bad Password 24	Bad Password 7 Days	Bad Username
Encryption Not Supported	Expired Password	Multiple Login Failures by IP Address	Pre-Auth required/Bad Password
Time Skew Too Great	Workstation Restriction		
NTLM			
Bad Password	Bad Password 24	Disabled	Expired
Failed Username	Failed Username 24	Locked Out	Out of Hours
Password Change	Password Expired	Workstation Restriction	

DAD Interface

This brings us to the piece of DAD that you will probably spend the most time with once it is up and running, the queries! DAD comes with a number of built in queries when you install it and the number of built in queries is constantly growing. In fact, if you develop any particularly useful queries, we strongly encourage you to post them in the DAD forums so that we might be able to include them for other DAD users!

You can find the DAD forums at <http://www.sourceforge.net/projects/lasseie>.

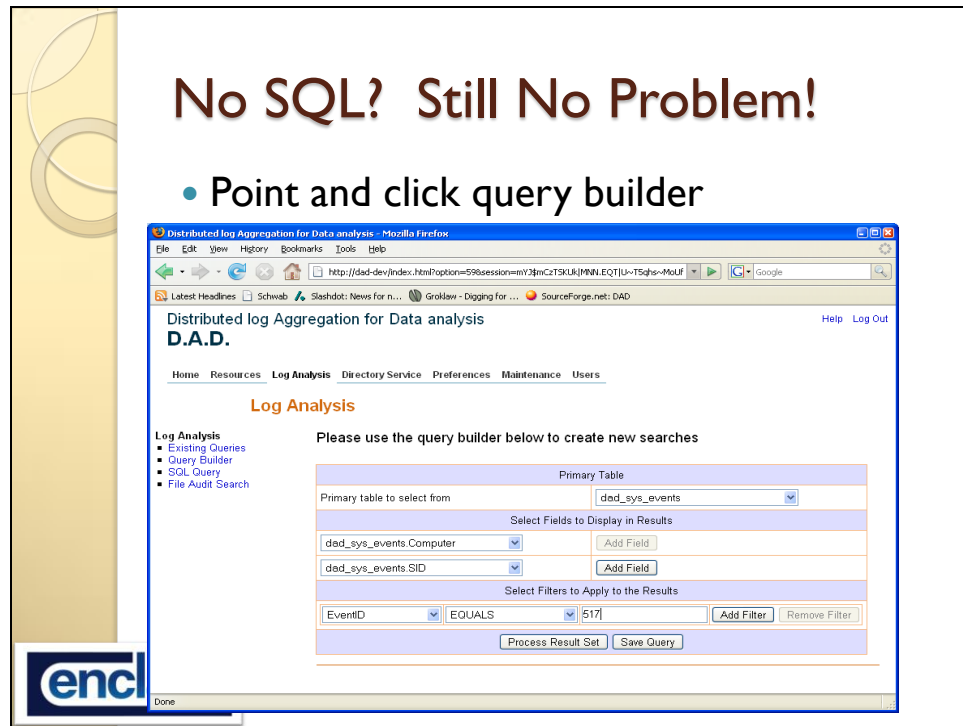


What if you can't find a query that satisfies your needs? No problem! If you are familiar with SQL, you can easily write any query that you like. Additionally, DAD exposes all of the tables within the DAD database so that you can perform joins across multiple tables with no additional effort.

While writing queries, you can make use of the "Process Query" button on the bottom of the SQL Query Maintenance page. This allows you to progressively model and refine your query and achieve instant satisfaction by seeing the results immediately below.

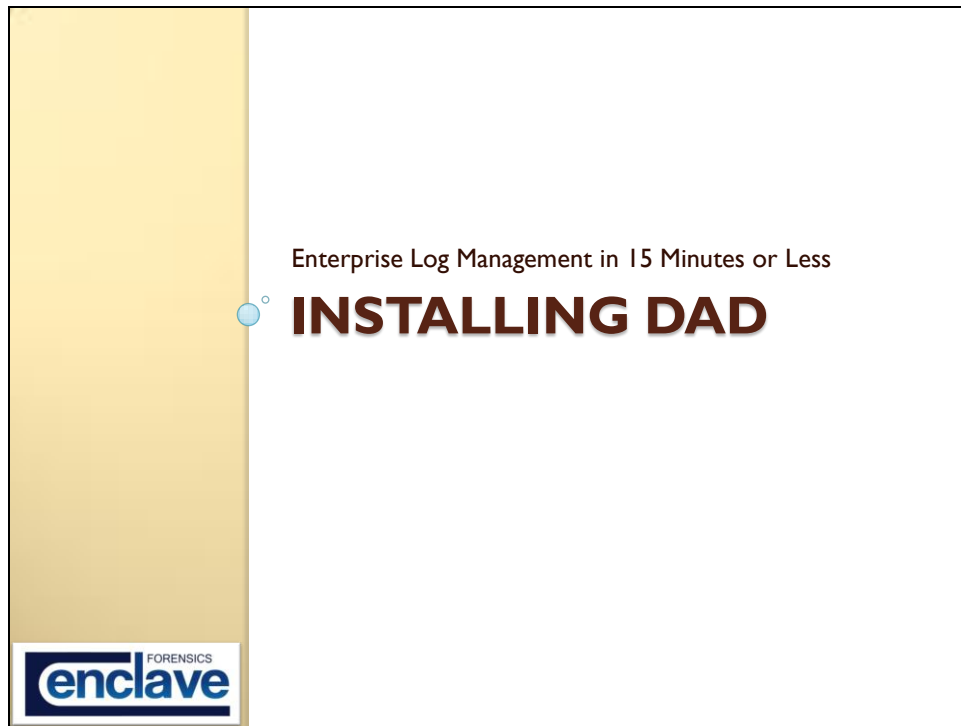
As a side note, when modeling queries we generally recommend that you append a LIMIT statement just in case you are asking for way more than you intended to. Another good strategy is to run a SELECT COUNT(*) with your selection criteria to verify how much data you are about to produce.

Once the query works just the way you want it to, you can optionally add the query to the Existing Queries page by giving it a name, a description and a category. You are not limited to the categories on the Existing Queries page. In fact, that page is built from a queries table and dynamically creates the categories. In the version of DAD that you are using, you likely also have the ability to select which roles should be permitted to execute this query. These appear as small checkboxes to the right of the query window.



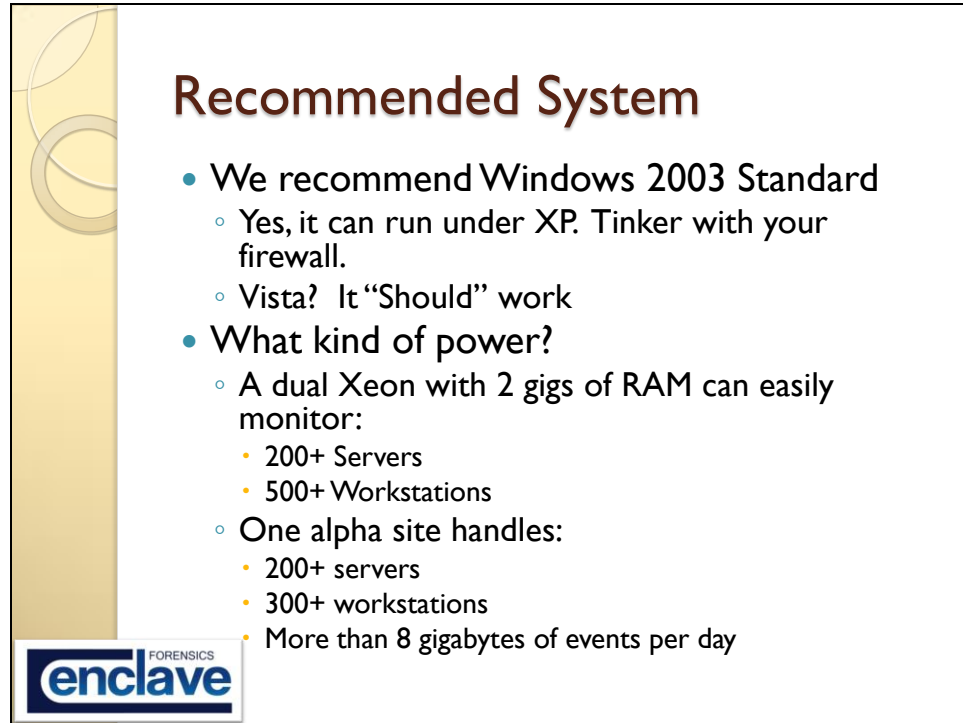
“Ah,” you say, “this means that I have to know how to write SQL queries?” The answer is that you actually do not have to know any SQL at all. In fact, there is a Query Builder that will allow you to select which tables to gather data from, which fields to include in the output and what criteria to match in the process. These queries can also be saved in the same way that hand crafted queries can.

At this point in time there is no plan to make the Query Builder any more intelligent than it is right now. The next step would be to allow table joins and other more advanced concepts. Quite frankly, if you plan to add a large number of queries and want them to be efficient, writing them by hand is absolutely the best way to go. The purpose of the Query Builder is really to allow for the creation of simple ad hoc queries.




Installing DAD

We've heard a great deal about how DAD functions. Now it's time to actually get DAD installed. At the time that this book was created we were hard at work trying to create an easy to use MSI installer for DAD to automate the entire installation process. You may find that this installer is available on your CD (then again, it may not be!). Even so, in class we are going to walk through the manual installation so that you can clearly see how the pieces fit together. This will make it easier for you to troubleshoot a DAD installation back at the office if the need arises.



Recommended System

- We recommend Windows 2003 Standard
 - Yes, it can run under XP. Tinker with your firewall.
 - Vista? It “Should” work
- What kind of power?
 - A dual Xeon with 2 gigs of RAM can easily monitor:
 - 200+ Servers
 - 500+ Workstations
 - One alpha site handles:
 - 200+ servers
 - 300+ workstations
 - More than 8 gigabytes of events per day



While we’re in class, your laptop will be more than sufficient for a trial installation of DAD. Back at the office, however, you will likely want to move this onto some actual server hardware. At one conference, before we had written such thorough directions on the installation of DAD, the joke was that all of the laptops had just become the official DAD servers for the respective organizations since the students did not feel confident that they could reproduce the installation on another host.


What should you use for server hardware? At this point, we haven’t really had the opportunity to perform extensive scalability tests. What we can give you some anecdotal guidance. If you choose to deploy DAD within your organization, we are very interested to hear how many systems you are monitoring, how many events you are storing and any performance bottlenecks that you experience so that we can expand on this knowledge.

At one site where DAD is deployed, a dual Xeon running at 2 GHz and with 2 gigabytes of RAM is monitoring more than 200 servers and more than 300 workstations simultaneously. This particular site performs extensive logging, generating more than eight gigabytes of event log data per day. In fact, two of the domain controllers roll their logs over every fifteen minutes during the peak hours of the day. Even with this load, this server is easily keeping up and can actually handle more data.



“Simply Install...”

- Copy “DAD” from the CD to C:\
- We need some supporting players:
 - MySQL
 - Apache
 - PHP
 - Perl
 - Java Runtime Environment



Please follow along with the instructor as he demonstrates how to install DAD manually. Detailed directions can be found below as well. These directions are intended for your use back at your office.

OS

Install server 2003. Standard edition is just fine. If 2003 Server is unavailable, you can certainly install DAD on a 2000 Server or even an XP or 2000 desktop, but the overall performance should be best on a Server platform.

We recommend that you set block size of data drive to 64k to allow for better space utilization for the database. We also recommend that the drive where the database files will reside be configured as a Dynamic Disk so that future expansion will be easily accomplished by simply adding drives to the volume set.

Unpacking DAD

DAD in its current incarnation comes as a ZIP file containing DAD itself, a starter data set and all of the software necessary to get DAD up and running. After your target server has been built, the first

task is to unpack the DAD suite. To do this, please unzip the DAD archive into C:\. When you have done this you should have a "C:\DAD" directory structure in which you will find a copy of this file, a "Source" folder, a "Web" folder and a "Jobs" folder. Once you have unpacked everything, you should progress through this document step by step.

If you choose to install DAD into some other source directory or drive, you **MUST** modify the "install_dad.bat" script to reflect the new location and any drive location changes.

NOTE: If you are installing on a 64 bit OS, you **MUST** change all of the paths in the installation files and in all of the configuration files that point to "C:\Program Files" to "C:\Program Files (x86)".

Installing DAD

The simplest way to perform the installation is to, after unpacking DAD into C:\DAD, start a command prompt, change to "C:\DAD\source" and run the "install_dad.bat" script. Simply follow the directions as prompted to complete the installation. We are *not* going to use this script to perform our installation in class.

MySQL

Using the MySQL installer provided, perform a typical install by accepting all of the defaults. It is not necessary to perform a MySQL.com Sign-Up during the installation though you are welcome to do so. When asked about configuration options, please be sure to configure MySQL to start as a service.

After completing the installation, it may generate an alert indicating that the service is not enabled or there is a firewall turned on. Ensure that the firewall is off and that the port (3306) is allowed to communicate. Sometimes the service is simply maybe slow to start, so you should 'retry' on the 'error' window.

Apache

Please use the Apache2 installer from the Apache directory. You can accept all of the default options throughout the installer.

PHP

Using the PHP 5 installer in the C:\DAD\Source\PHP directory, please perform a standard installation of PHP after selecting "Apache" as the target server type, accepting all of the defaults. You may, of course, adjust the email address as you see fit when prompted. At the end of the installation

you will likely see a message indicating that the installer cannot configure the web server despite asking if you wish for it to do so.

Perl

Using the installer in the C:\DAD\Source\Perl directory, perform a default installation of Perl. After the installation has completed successfully, please open a command prompt. At the command line, please type:

```
cd \DAD\Source\Perl
.\InstallModules
```

The "InstallModules" script will automatically perform the installation of the remaining necessary modules for Perl and DAD to operate correctly. If you would like more detail, the script is installing the following:

- DBI.ppd
- DBD-mysql.ppd
- DBD-ODBC.ppd
- Convert-ASN1.ppd
- perl-lda.ppd
- Win32-EventLog.ppd
- GD
- GDtextutil
- GDGraph

EXTREMELY IMPORTANT:

As distributed, DAD has a default password of "All4Fun" configured as the password for the 'root' user in the MySQL database. If you wish to get up and running ASAP, you might wish to set this as the password for your root user on your SQL server. Please note that we do not -recommend- this, we simply are pointing out that you might choose to do this. What we recommend is that you choose a good password and then create another user for DAD to operate with. This second account should only have rights to the DAD database and tables.

After the installation has been completed, you MUST modify the C:\DAD\Web\Config\dbconfig.php file to reflect the correct IP address (should you choose to install the SQL service on another server), the DAD username used to access the database and the password used by this user.

You MUST also edit the C:\DAD\jobs\Log Parser\Aggregator.ph file. In this file you will find these lines:

```
$MYSQL_SERVER="localhost";  
$MYSQL_USER="root";  
$MYSQL_PASSWORD="All4Fun";
```

You must adjust the server, user and password appropriately to allow the aggregator to access the database.

Post Installation

Once DAD is installed, there are a few things that you'll need to do to get up and running. For this initial release, this is a manual process but in future releases this will be better automated. First off, you will need to either create an account under which the DAD aggregator will run or decide whose credentials DAD will use. Once you have done this, we recommend that you use group policy in your domain to grant this account the "Manage Security and Audit Logs" right to computers in the domain. In most installations we would recommend applying this right for all computers, domain servers and domain controllers. For class today, we will simply run DAD under our own credentials.

Please note: If you wish to import logs other than the Security log, the simplest solution is to add the service account to the local administrators group for all of the systems that are monitored. There are more difficult ways to add this capability without making the service account an administrator. We will include details on how to configure these access control lists for Windows logs later today.

```
cd \DAD\jobs\Log Parser
```

```
.\aggregator.pl
```

Troubleshooting

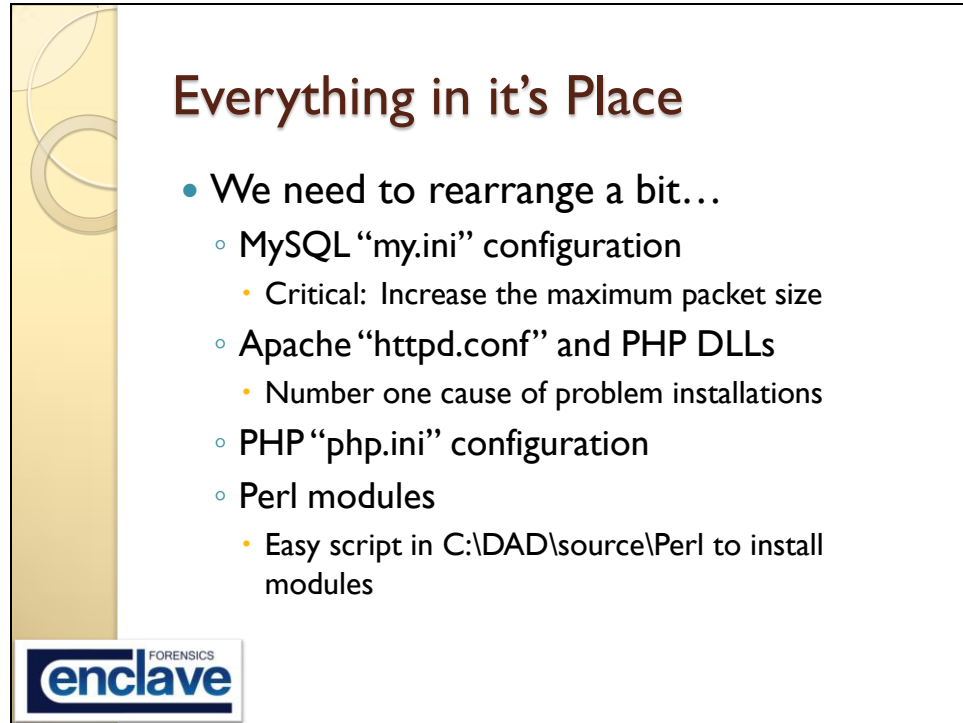
1) Apache error log reports "PHP Warning: PHP Startup: Unable to load dynamic library

'C:\PHP\php_mysql.dll' - The specified module could not be found.\r\n in Unknown on line 0"

This error, when the `php_mysql.dll` is in fact in the `C:\PHP` directory (and the correct version), is very misleading. The actual problem is that "`libmysql.dll`" is not in the `Apache\bin` directory (or otherwise in the Apache path).


2) I've added a workstation (or workstations) to the list of systems to collect data from but nothing seems to be entering the database from these systems. What's wrong?

This problem is caused, strangely enough, by the Windows Firewall. Unfortunately, there is no simple granular control to change this behavior. The simplest resolution is to either disable the firewall (not recommended unless you have some other firewall on there too) or to simply enable File Sharing in the Advanced settings. We're not actually sharing files, but this seems to take in all sorts of SMB communication including the logs. In the future, if we have the energy, we'll describe more specifically how this can be accomplished through Group Policy to only allow the DAD server to collect logs without completely opening up file sharing. Of course, opening the file sharing ports doesn't actually share anything; it just makes the connections possible.



Everything in it's Place

- We need to rearrange a bit...
 - MySQL "my.ini" configuration
 - Critical: Increase the maximum packet size
 - Apache "httpd.conf" and PHP DLLs
 - Number one cause of problem installations
 - PHP "php.ini" configuration
 - Perl modules
 - Easy script in C:\DAD\source\Perl to install modules



Configuring DAD

Even though everything is installed, we still need to move some files around to configure everything correctly.

MySQL

Once the MySQL server is up and running, please open a command prompt. At the command prompt please enter:

```
sc stop mysql
```

This will halt the MySQL server. This is necessary since we will be replacing the configuration file and relocating the database data files. Once you have stopped the service, please copy the "my.ini" file from the \DAD\Source\MySQL directory to c:\Program Files\MySQL\MySQL Server 5.0\. This will replace the MySQL configuration built by the installer. Additionally, unless you choose to edit the "my.ini" file to store your SQL data files in another location, you MUST move the "data" directory from the MySQL installation directory into "C:\DAD\". Once you have completed these two tasks, please type the following at the command prompt to restart the MySQL Server:

sc start mysql

Apache

Once the installer has completed, please copy the "httpd.conf" file from the C:\DAD\Source\Apache directory into "**c:\Program Files\Apache Group\Apache2\conf**". You must also copy the "libmysql.dll" file from the Apache Source directory into the Apache2\bin directory in order to allow PHP to correctly interact with the MySQL database.

PHP

After completing the PHP installation, please copy the following files from the C:\DAD\Source\PHP directory to the following locations:

copy **libmysql.dll** to **C:\Program Files\Apache Group\Apache2\bin**

copy **php_mysql.dll** to **c:\php**

copy **php_mysql.dll** to **c:\php**

copy **php5apache2.dll** to **c:\php**

copy **php_gd2.dll** to **c:\php**

delete c:\windows\php.ini

copy **PHP.ini** to **c:\php**


Please left-click on the Apache icon in the icon tray and restart the Apache server now.

All Hands Accounted for

- With everything installed:
 - “sc restart mysql”
 - Use “sc query mysql” to make sure it restarted
 - Restart Apache
 - If it won’t start, there is a “Test Configuration” option
- You should be able to pull up a page at <http://localhost>
 - The page is not yet complete, but it should at least say “DAD” on it




At this point, if you have followed the directions carefully, you should be able to use your web browser to connect to localhost. When you connect, you will not actually get a login screen. This is no problem. If, however, you cannot connect at all, please ask the instructor for assistance now!



Schema + Data = DAD!

- We still need to import the Schema
- We also need some starter
 - Menus, default queries, starter user, etc.
 - Script in source directory
 - “populate_database <host>”
 - It will prompt you for your MySQL Password
- Now you should really have a login page
 - User: admin Password: Password1



Final Installation

At this point, you have all of the software in place and DAD is just about ready to use. To make it fully functional, we need to install the Database schema and some put some initial data into the tables. To do this, please change into the "C:\DAD\Source" directory.

In the Source directory you will find two files: Creates.sql and Starter_Data.sql. Please open a command prompt and change to the C:\DAD\Source directory. Type the following to load this data into your database:

```
"c:\Program Files\MySQL\MySQL Server 5.0\bin\mysql.exe" -u root -p < Creates.sql
```

MySQL will prompt you for the password that you selected during installation of the SQL server engine. Please type your password now. After this completes please type:

```
"c:\Program Files\MySQL\MySQL Server 5.0\bin\mysql.exe" -u root -p -D DAD < Starter_Data.sql
```

Once these steps are completed, you should be able to successfully authenticate into the DAD application using a username of “admin” and a password of “Password1”.

The Scheduler Service

- The next step on a server would typically be to install the Scheduler service
 - There's a script to do this for you
- Let's not install a new service on your laptop
 - Instead, let's kick off the aggregator manually

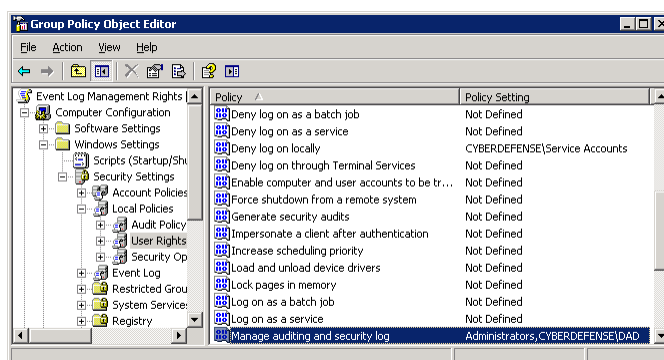


In a typical installation, the next step would be to install the scheduler service. Within the source directory there is a script that will do this for you. For our laptops, however, we would prefer not to install any more background services, especially since your laptop likely will not remain the DAD server for your company.

Rather than starting the scheduler, we can start the aggregator manually. To do so, simply browse to the C:\DAD\jobs\log parser directory and double click on the aggregator.pl file. At this point you should see the aggregator start up and wait for logs.

Aside: Domain Configuration

- Aggregator needs rights to poll event logs
 - Create a group policy at the appropriate OU to grant “Manage Auditing and Security Log”




Domain Configuration

It is important for us to take a few moments at this point to explain what else will change when installing DAD into a domain. Since the aggregator will be running with our local or cached credentials on our laptops it is quite likely that the aggregator will have no problem pulling the logs from your laptop. Within your domain, however, we need to do a few things to make this happen.

For a Windows 2000 domain, the only thing that we need to do is to create a group policy that grants the DAD user account the “Manage auditing and security log” right. This right allows the DAD account to read, write and even clear the security log. The other logs are automatically accessible under Windows 2000 and lower.

If you are using a Windows 2003 domain, there is a little bit more work required.

In Windows 2003 and higher domains the easy way to get access to these other logs is to put the DAD account into the Domain Administrators or Enterprise Administrators group, but that would be very bad security practice. The right way to do it is to create a SDDL DACL on each of the logs that we are interested in and to push this out through group policy.



Simplified SDDL

- We need to grant read access to the various logs
 - O:BAG:SYD:(D;;0xf0007;;;AN)(D;;0xf0007;;;BG)(A;;0xf0007;;;SY)(A;;0x7;;;BA)(A;;0x7;;;SO)(A;;0x3;;;IU)(A;;0x3;;;SU)(A;;0x3;;;S-1-5-3)(A;;0x1;;;DAD's SID Goes Here)
- Great explanation:
 - <http://www.leastprivilege.com/EventLogACLsInWindows2003.aspx>
- Adding log DACLS to Group Policy:
 - <http://support.microsoft.com/default.aspx?scid=kb;en-us;323076>

SDDL, to say the very least, is esoteric. We found a few nice resources that explain SDDL and we've extracted them below. The first comes from www.leastprivilege.com:

SDDL Explained

Under Windows 2003 the EventLogs are ACLed - they can be found in the registry, e.g. HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Eventlog\Application\CustomSD

The default ACL for the Application Log is:

```
O:BAG:SYD:(D;;0xf0007;;;AN)(D;;0xf0007;;;BG)(A;;0xf0007;;;SY)(A;;0x7;;;BA)(A;;0x7;;;SO)(A;;0x3;;;IU)(A;;0x3;;;SU)(A;;0x3;;;S-1-5-3)
```

Nice, eh?? This is SDDL (Security Descriptor Description Language) and means:

- O:BA Object owner is Built-in Admin (BA)
- G:SY Primary group is System (SY)
- D: This is a DACL, rather than an audit entry or SACL
- (D;;0xf0007;;;AN) Deny Anonymous (AN) all access
- (D;;0xf0007;;;BG) Deny Built-in Guests (BG) all access

- (A;;0xf0005;;;SY) Allow System Read and Clear, including DELETE, READ_CONTROL, WRITE_DAC, and WRITE_OWNER (indicated by the 0xf0000)
- (A;;0x7;;;BA) Allow Built-in Admin READ, WRITE and CLEAR
- (A;;0x7;;;SO) Allow Server Operators READ, WRITE and CLEAR
- (A;;0x3;;;IU) Allow Interactive Users READ and WRITE
- (A;;0x3;;;SU) Allow Service accounts READ and WRITE
- (A;;0x3;;;S-1-5-3) Allow Batch accounts (S-1-5-3) READ and WRITE

If you want to extend that ACL - you have to add the SID of the account and an access mask, which is as follows:

- 0x0001 ELF_LOGFILE_READ Permission to read log files.
- 0x0002 ELF_LOGFILE_WRITE Permission to write log files.
- 0x0004 ELF_LOGFILE_CLEAR Permission to clear log files.

The second reference that we have included is quoted from Microsoft itself. This reference explains how you can add references for the log DACLs to your group policy management interface:

Adding DACLs to Group Policy

Important: To view the group policy settings that are described in this article in the Group Policy editor, first complete the following steps, and then continue to the "Use Group Policy to Set Your Application and System Log Security" section:

Use a text editor such as Notepad to open the Sceregvl.inf in the %Windir%\Inf folder. Add the following lines to the [Register Registry Values] section:

- MACHINE\System\CurrentControlSet\Services\Eventlog\Application\CustomSD,1,%AppCustomSD%,2
- MACHINE\System\CurrentControlSet\Services\Eventlog\Security\CustomSD,1,%SecCustomSD%,2
- MACHINE\System\CurrentControlSet\Services\Eventlog\System\CustomSD,1,%SysCustomSD%,2
- MACHINE\System\CurrentControlSet\Services\Eventlog\Directory Service\CustomSD,1,%DSCustomSD%,2
- MACHINE\System\CurrentControlSet\Services\Eventlog\DNS Server\CustomSD,1,%DNSCustomSD%,2
- MACHINE\System\CurrentControlSet\Services\Eventlog\File Replication Service\CustomSD,1,%FRSCustomSD%,2

Add the following lines to the [Strings] section:

- AppCustomSD="Eventlog: Security descriptor for Application event log"
- SecCustomSD="Eventlog: Security descriptor for Security event log"
- SysCustomSD="Eventlog: Security descriptor for System event log"
- DSCustomSD="Eventlog: Security descriptor for Directory Service event log"
- DNSCustomSD="Eventlog: Security descriptor for DNS Server event log"
- FRSCustomSD="Eventlog: Security descriptor for File Replication Service event log"


Save the changes you made to the Sciregvl.inf file, and then run the regsvr32 scecli.dll command.

Start Gpedit.msc, and then double-click the following branches to expand them:

- Computer Configuration
- Windows Settings
- Security Settings
- Local Policies
- Security Options


View the right panel to find the new "Eventlog" settings.

1. In the Active Directory Sites and Services snap-in or the Active Directory Users and Computers snap-in, right-click the object for which you want to set the policy, and then click Properties.
2. Click the Group Policy tab.
3. If you must create a new policy, click New, and then define the policy's name. Otherwise, go to step 5.
4. Select the policy that you want, and then click Edit. The Local Group Policy MMC snap-in appears.
5. Expand Computer Configuration, expand Windows Settings, expand Security Settings, expand Local Policies, and then click Security Options.
6. Double-click Event log: Application log SDDL, type the SDDL string that you want for the log security, and then click OK. Repeat for each log that you wish to set a DACL for.



Now What?

- DAD is working but has nothing to do
 - Adding Servers
 - Working with Queries
 - Creating queries
 - Future schema plans
 - Working with Users
 - Roles and what they apply to
 - Extending the Architecture
 - Menu administration



Adding Servers and Computers to Process

The final step in setting up DAD is to add systems to the list of servers to aggregate logs from. To add servers and workstations, log in as an administrator. Under the “Maintenance” tab, select the “Systems” option. Enter all of the pertinent information including, most importantly, the NETBIOS name of the system that you wish to add. Please also include a priority value of 1 or higher. A zero will disable collection from the host. Lower numbers (1, for instance) indicate higher priority. Typically, we recommend setting DCs and Global Catalog servers as priority 1. Domain Servers would be priority 2. Everything else would be priority 3 or more. Once you have entered the necessary information, be sure to select which logs you wish to collect from this system. Finally, click on the “Save as New” button to begin collections from that system.


Please Note: When you first add a system to aggregate logs from, the aggregator may report a SQL error. This is **expected** and can be safely ignored. This error will be eliminated in future releases.

What About the Log Carver?

- Requires that we know a bit about Regex
 - Interface provides an “Explain” function
 - Copy and paste in sample logs to test carver
- How about a gentle introduction to regular expressions?




At this point, let's turn up the heat a little bit and go through a gentle introduction to regular expressions. This will allow us to create our own custom log carving signatures. Depending on connectivity, we can even experiment with some live data off of a real domain to see if we can create parsing expressions together.



Regex Basics

- Using Meta Characters to Represent Data
 - Wildcards at a command line are an example

<ul style="list-style-type: none"> • Match any character ? Match exactly one of the last expression + Match one or more of the last expression * Match zero or more of the last expression ^ Match the beginning of the line 	<ul style="list-style-type: none"> \$ Match the end of the line [] Match a set of characters [^] Match everything except the set {n} Match the last expression exactly n times {n,} Match the last expression n or more times {n,m} Match the last expression at least n times but not more than m times
---	--



Regular Expressions

Regular expressions are based on what are known as meta characters. This essentially means characters that are used to represent or describe other characters. This is very similar to using an asterisk at the command line to represent filenames. Of course, there's a lot more to it than an asterisk! Here's a basic chart that lists the most useful meta characters that we will use:

.	Matches exactly one of any character.
?	Matches exactly one of the previous pattern. For instance "a?" will match an 'a' followed by exactly one 'a'.
+	Matches one or more of the previous pattern. For instance "a+" will match an 'a' followed by one or more 'a's.
*	Matches zero or more of the previous pattern. For instance, "a*" will match zero or more 'a's.
^	The caret will match the beginning of the line. For instance "^The quick" will match "The quick" but only when it appears at the beginning of the line.
\$	Matches the end of the line. For instance, "lazy dog\.\$" will match "lazy dog." but only when it appears at the end of the line.

[]	Allows you to specify a range or group of characters. For instance, “[a-z]” will match one lower case roman letters. “[abc123]” will match one character from the set described between the brackets.
[^]	When the caret appears as the first character in a group it inverts the group. For example the group “[^a-z]” will match one character that is anything <i>except</i> for a lower case roman character.
{n}	The curly braces are used to repeat matches <i>n</i> number of times for the last expression. For example, “[a-z]{3}” will match exactly three lower case roman characters.
{n,}	When the number within the braces is followed by a comma, the previous expression will be matched at least <i>n</i> times. For example, “[a-z]{3,}” will match three or more lower case roman characters.
{n,m}	When the braces contain two numbers separated by a comma the previous expression is matched between <i>n</i> and <i>m</i> times, inclusive. For example, “[a-z]{3,6}” will match at least three lower case roman characters but not more than six.

Using these building blocks we can represent any kind of text based data that might come our way. As you can imagine, these expressions can become rather complicated. We will examine some sample expressions on the next few pages before we start writing some of our own.

Whew. What Does That Mean?

- Try some examples:

```
Apr 9 14:17:01 localhost CRON[19673]: (pam_unix) session opened for user root by (uid=0)
Apr 9 14:17:01 localhost CRON[19673]: (pam_unix) session closed for user root
Apr 9 14:27:29 localhost sshd[19675]: Connection from 192.168.254.1 port 4365
Apr 9 14:27:30 localhost sshd[19675]: Failed none for dhoelzer from 192.168.254.1 port 4365 ssh2
Apr 9 14:27:35 localhost sshd[19675]: Accepted keyboard-interactive/pam for dhoelzer from 192.168.254.1 port 4365 ssh2
Apr 9 14:27:35 localhost sshd[19679]: (pam_unix) session opened for user dhoelzer by dhoelzer(uid=0)
```

- Which lines will these match/not match:

- `[A-Z][a-z]{2}.*sshd\[([0-9]+)\]`
- `.*localhost.*for user`
- `.*Failed.*`



Let's try out some examples. To do so we will work with the following sample text:

```
Apr 9 14:17:01 localhost CRON[19673]: (pam_unix) session opened for user
root by (uid=0)

Apr 9 14:17:01 localhost CRON[19673]: (pam_unix) session closed for user
root

Apr 9 14:27:29 localhost sshd[19675]: Connection from 192.168.254.1 port
4365

Apr 9 14:27:30 localhost sshd[19675]: Failed none for dhoelzer from
192.168.254.1 port 4365 ssh2

Apr 9 14:27:35 localhost sshd[19675]: Accepted keyboard-interactive/pam for
dhoelzer from 192.168.254.1 port 4365 ssh2

Apr 9 14:27:35 localhost sshd[19679]: (pam_unix) session opened for user
dhoelzer by dhoelzer(uid=0)
```

Consider the various expressions listed in the slide. See if you can predict which lines in the above log will be matched by the various regular expressions given. The answers are on the next page, but no cheating!

[A-Z][a-z]{2}.*sshd\[([0-9]+)\]

Apr 9 14:27:29 localhost sshd[19675]: Connection from 192.168.254.1 port 4365

Apr 9 14:27:30 localhost sshd[19675]: Failed none for dhoelzer from 192.168.254.1 port 4365 ssh2

Apr 9 14:27:35 localhost sshd[19675]: Accepted keyboard-interactive/pam for dhoelzer from 192.168.254.1 port 4365 ssh2

Apr 9 14:27:35 localhost sshd[19679]: (pam_unix) session opened for user dhoelzer by dhoelzer(uid=0)

.*localhost.*for user

Apr 9 14:17:01 localhost CRON[19673]: (pam_unix) session opened for user root by (uid=0)

Apr 9 14:17:01 localhost CRON[19673]: (pam_unix) session closed for user root

Apr 9 14:27:35 localhost sshd[19679]: (pam_unix) session opened for user dhoelzer by dhoelzer(uid=0)

.*Failed.*

Apr 9 14:27:30 localhost sshd[19675]: Failed none for dhoelzer from 192.168.254.1 port 4365 ssh2


Ok, That's Not so Bad...

- But wait, there's more!
 - We need to select pieces of the output and carve it up
 - DAD currently supports 25 arbitrary fields plus the system and some miscellaneous data
- We can select pieces using the parenthesis characters
 - Each selection is numbered



Using regular expressions to match data does take some thinking, but as you can see it's really not that tough. Even though we've managed to match lines, though, we still need to know how to cut that data up into something useful. Remember that the data that we are extracting needs to be inserted into a database. For this reason, the data must be parsed into fields that make sense (at least to us) so that it can be indexed for later searching.

With regular expressions it is possible to do this using the parenthesis characters. When you surround pieces of your regular expression with parentheses, it creates what are known as "backreferences". Actually, what you are doing is marking pieces of the data with your expression so that they can later be referred to using backreferences. With standard regular expressions you are limited at nine backreferences. We are using Perl extended regular expressions which will afford us more references than we actually need.




Let's try one

Apr 9 14:27:30 localhost sshd[19675]: Failed none for dhoelzer from 192.168.254.1 port 4365 ssh2

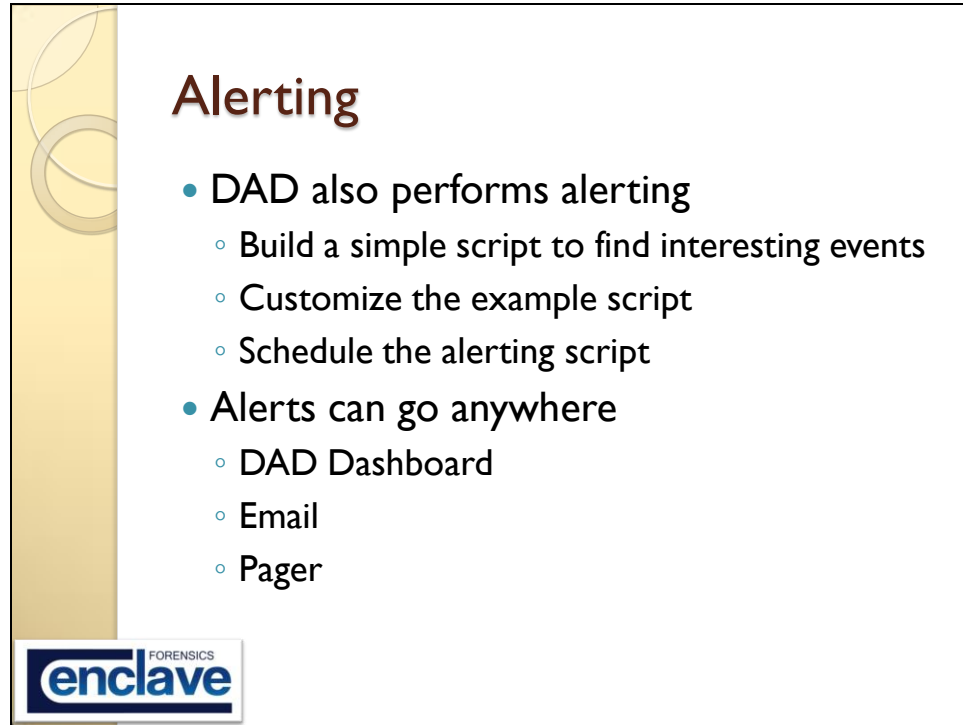
- First make a regex that works:
 - `.*sshd\[0-9]+\:` .+ for .+ from .+ port [0-9]+ .+
- Once that works, add selectors
 - `.*(sshd)\[0-9]+\: (.+) for (.+) from (.+) port ([0-9]+) (.+)`
 - This produces:

```
sshd Failed none dhoelzer 192.168.254.1 4365 ssh2
sshd Accepted keyboard-interactive/pam dhoelzer 192.168.254.1 4365 ssh2
sshd Accepted keyboard-interactive/pam dhoelzer 192.168.254.1 4365 ssh2
sshd Accepted keyboard-interactive/pam dhoelzer 192.168.254.1 4365 ssh2
sshd Accepted keyboard-interactive/pam dhoelzer 192.168.254.1 4365 ssh2
```




Generally speaking, unless you are a regular expression expert, it is easier to begin by creating an expression that matches the lines in the data that you are looking for. This is actually the harder part of the process. As we experiment with regular expressions you will likely understand more fully why regular expressions take some time to master and why two people can come up with completely different regular expressions that do essentially the same thing.

Once you have a working expression, simply mark off the pieces of data that you are interested in with parentheses. Behind the scenes, these selectors are used by DAD to split the data out into fields that can be inserted into the database. You can see in the slide pictured above the progress from an expression to an expression with selectors and finally the output that can be produced once the selectors are in place.



Alerting

- DAD also performs alerting
 - Build a simple script to find interesting events
 - Customize the example script
 - Schedule the alerting script
- Alerts can go anywhere
 - DAD Dashboard
 - Email
 - Pager




Alerting

Another DAD feature that we are sure that you will be interested in is Alerting. Alerting is still in its infancy in DAD. Still, because of the extensibility of the architecture, you can create extremely complex correlation and alerting functions.


DAD includes several example sample scripts that all create an alert for the same type of issue. These samples are provided in Perl, PHP and Java. If you wish to create a custom alert, you can really use any language that can be run from the DAD scheduler. These files are provided to make it easier for someone new to DAD to see how an alert can be generated and how that data can be populated onto the dashboard.

Alerts can actually do anything. In fact, they don't even need to query the DAD database. You could, for instance, create an alert that tests to see if a host is up and then report its status to the DAD dashboard, send an email to alert someone that the host is down or even send a message through a pager gateway. You can really do anything that you set your mind to.




What DAD is Not

- DAD is not a...
 - ...System Health Monitor
 - ...System Management Tool
 - ...Centralized Configuration Tool




We would like to take just a few moments to explain what DAD is not. DAD is clearly not a SEIM. DAD is not a health management tool for your systems. DAD is not designed to alert you to performance or load problems. DAD is not an enterprise systems management tool. DAD is not a configuration management tool. DAD is not a patch management solution.

Not only does DAD not do these things, DAD will likely never take on these roles (unless someone wants to write modules that do these things and give them to us!) While interesting functionality may be added to DAD in the future, the tool exists to satisfy only one major issue: aggregating, managing, correlating and alerting on log events.



How Can I Scale This?

- Deployments for a few hundred systems work fine on a single box
- Scalability options:
 - Deploy a database cluster
 - Deploy a standalone aggregator
 - Increase the thread counts on the aggregator
 - Deploy a standalone scheduler/alerter



Scalability

One other last issue before we wrap up. People wonder what they can do if they want to scale DAD for large enterprises. Before you start doing any of the things listed below, we would suggest that you first install DAD on server class hardware and try it out. If you find that the volume of information that you are trying to handle is too great then you might consider doing some of the following things.

The first and likely most dramatic performance gain can be realized by creating a separate database server. In almost all cases that we have seen, the biggest performance bottleneck in DAD can be getting very busy logs into the database quickly. Moving the database to a dedicated database server with a large amount of memory and tweaking the configuration of the database to leverage these improvements can do much to relieve this load. If even greater performance is required, migrating to a database cluster will also show tremendous benefits. This can be a pricey solution.

If you find that your system bottleneck is coming from having many alerts running simultaneously, then the next step would be to divorce the scheduler from the DAD server. The scheduler can run anywhere as long as it is configured to speak to the database server. In an environment with many and frequent alerts this can provide a good performance boost.

It is also possible to move the aggregator to its own stand alone system. This would only be recommended in rare instances, perhaps where ten thousand or more hosts are being monitored. If you need to go this route then it would be wise to adjust the number of threads available to the aggregator within the aggregator.ph file in the C:\DAD\jobs\log parser directory. If you are considering taking this step then you should have already created a standalone database server, perhaps even a cluster.

DAD Futures

- Continued focus on
 - Interesting alerts
 - Generally useful queries
 - Expanding Syslog carving library
 - Better handling of large results sets
- Longer Term
 - Schema revamp
 - Localization

For More Information

- www.cyber-defense.org
 - Open source arm of Enclave Forensics
- Available for download from Sourceforge
 - www.sourceforge.net/projects/lassie
 - If you're interested, at the least sign up for the Announcements list
 - Very low volume
 - Regular (almost daily) SVN updates are available



° **QUESTIONS?**