where information lives®

Enterprise Standards and Automation for Storage Integration and Installation

EMC Proven Professional Knowledge Sharing 2009

Aaron Baldie,
Enterprise ATC @ Microsoft
Baldie_Aaron@emc.com
EMC

Enterprise Standards and Automation for Storage Integration and Installation

Aaron Baldie
Enterprise ATC @ Microsoft

Baldie_Aaron@emc.com

EMC²

Table of Contents

Summary

It is daunting to keep systems up to date and in compliance with the latest drivers and software stacks. This is particularly true when working with over 2700 applications (Pohto, 2009) running on CLARiiON® and Symmetrix® DMX in the Microsoft production environment across thousands of servers. Microsoft leverages a number of processes to accomplish this task with web-based patching and updating technologies ranging from Windows updates to vendor supplied automation scripts for hardware specific installs. EMC provides the latter by working with our partners to develop install standards that can be deployed across every corporation. We have created a consolidated deployment package with the help of Dell, Emulex, Qlogic, and Brocade to keep the thousands of storage area network (SAN) attached servers up to date using standards based on the extensive work done with the EMC support matrix. This helps to deliver greater uptime to companies' top tier applications.

Leverage your partners

In this economic climate, a good vendor will be a great partner. They are leaving the majority of the heavy lifting on your shoulders if they sell to you and walk away. Drivers, applications and multi-vendor compatibility don't automagically appear after the product delivery. A certain amount of research and testing is needed to make sure that a deployed application works on the first try.

First impressions are everything in IT. If you have put your reputation on the line by purchasing a storage array to help with your companies growing needs, you want your vendor standing by you from installation to go live. Not only will this ensure a successful launch, but you will be in good hands if things do not go as planned.

For Microsoft's automated installation kit, I worked with my partners Emulex, Qlogic and Brocade to understand how their host bus adapter (HBA) packages worked in user mode and in unattended or silent mode. The install packages did not always work as expected so we worked together to fix the issue and try again. It was helpful to me and to the company to work with partners to resolve an issue. The bugs and pitfalls experienced during testing at Microsoft made the products better for every one of my partner's customers.

Providing hardware for testing is another way that companies extend their partnerships with EMC and Microsoft. This allows Microsoft to test product features while also integrating the products into the broader application and compute environment. With the hardware onsite and installed in an environment similar to production, the staff at Microsoft can work through any unknowns that arise to develop a solid deployment plan when released to production. The other benefit is the ability to test new features or next generation products and evaluate their merits to guide informed decision -making.

Standards

It would be nearly impossible to keep systems current or even know what to update in an environment without published standards. Standards can be referenced if a new issue is found in the production environment. Once a standard is established, it should be locked in for a pre-determined timeframe at which time it should be updated. Archive the old standard as N-1. At Microsoft, this is a 6 month cycle with support for N-1 systems continuing for 6 more months. This allows the standard to be on campus for one year.

What should be included in a published standards document? For a SAN attached host, there are five basic software components needed in a Windows environment.

1. HBA driver
2. HBA administration software
3. Multi-path software
4. Windows storage driver (aka storport)
5. Agents for the storage

These five software components must be compatible with what they are attaching to. In a SAN, we have to make sure that the HBA firmware, fabric switches and storage arrays are also up to date.

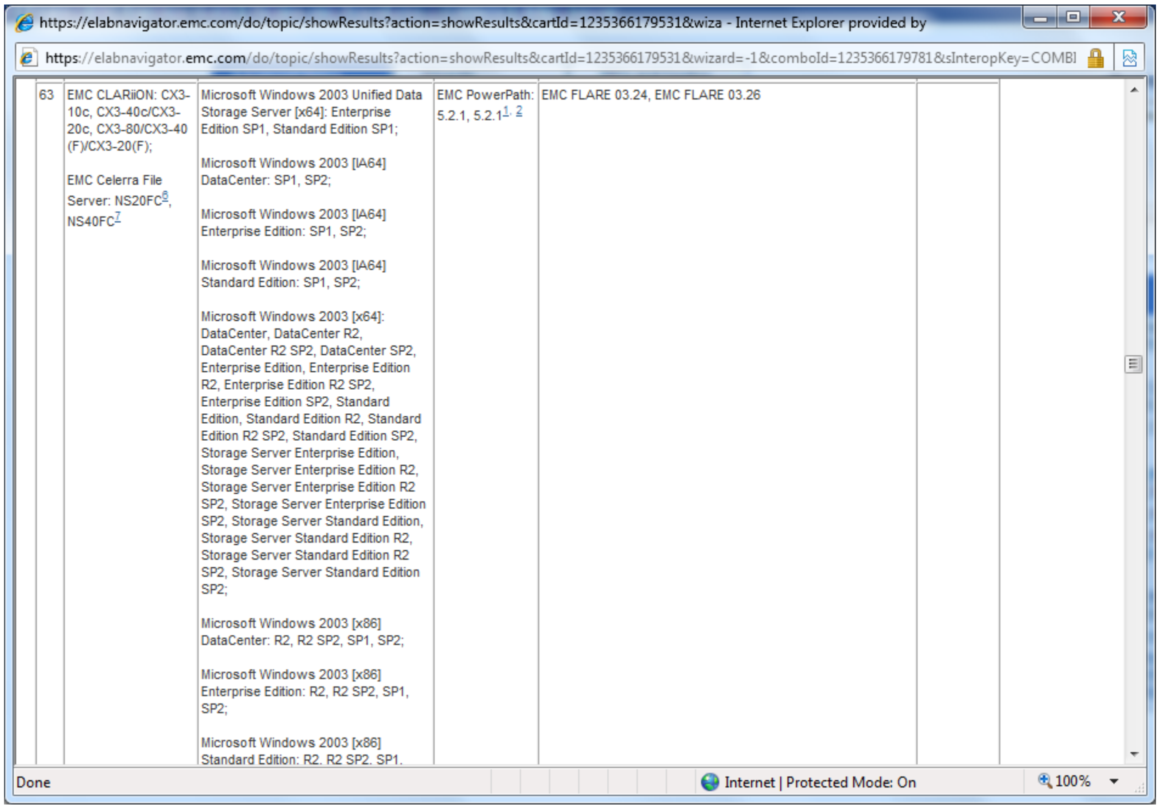
The published standard for a SAN will include all of the items outlined above and detailed below.

- Arrays firmware
    - Enginuity for Symmetrix DMX
    - FLARE® for CLARiiON
- Switch firmware
    - Brocade
    - Cisco
- HBA firmware
    - Emulex
    - QLogic
    - Brocade
- OS
    - HBA driver
    - HBA administrative software
    - Multi-path software
        - MPIO
        - PowerPath®
    - Windows storage driver (aka storport)
    - Agents for the storage
        - NaviAgent
        - Solution Enabler

This is only an example for Symmetrix DMX and CLARiiON in a Windows environment, but there are many ways to determine compatibility. A customer would need a full time person to handle the testing and regression of every possible permutation. So how do you keep up? This is where the EMC support matrix and eLabs come in to play.

EMC Support Matrix

Please take this opportunity browse https://elabnavigator.emc.com. It is amazing how much information the Support Matrix contains and what you can find in just a few clicks.


https://elabnavigator.emc.com/do/topic/showResults?action=showResults&cartId=1235366179531&wiza - Internet Explorer provided by

https://elabnavigator.emc.com/do/topic/showResults?action=showResults&cartId=1235366179531&wizard=-1&comboId=1235366179781&sInteropKey=COMBI

63	EMC CLARiiON: CX3-10c, CX3-40c/CX3-20c, CX3-80/CX3-40 (F)/CX3-20(F);
	
	EMC Celerra File Server: NS20FC[6], NS40FC[7]
		Microsoft Windows 2003 Unified Data Storage Server [x64]: Enterprise Edition SP1, Standard Edition SP1;
		
		Microsoft Windows 2003 [IA64] DataCenter: SP1, SP2;
		
		Microsoft Windows 2003 [IA64] Enterprise Edition: SP1, SP2;
		
		Microsoft Windows 2003 [IA64] Standard Edition: SP1, SP2;
		
		Microsoft Windows 2003 [x64]: DataCenter, DataCenter R2, DataCenter R2 SP2, DataCenter SP2, Enterprise Edition, Enterprise Edition R2, Enterprise Edition R2 SP2, Enterprise Edition SP2, Standard Edition, Standard Edition R2, Standard Edition R2 SP2, Standard Edition SP2, Storage Server Enterprise Edition, Storage Server Enterprise Edition R2, Storage Server Enterprise Edition R2 SP2, Storage Server Enterprise Edition SP2, Storage Server Standard Edition, Storage Server Standard Edition R2, Storage Server Standard Edition R2 SP2, Storage Server Standard Edition SP2;
		
		Microsoft Windows 2003 [x86] DataCenter: R2, R2 SP2, SP1, SP2;
		
		Microsoft Windows 2003 [x86] Enterprise Edition: R2, R2 SP2, SP1, SP2;
		
		Microsoft Windows 2003 [x86] Standard Edition: R2, R2 SP2, SP1,
			EMC PowerPath: 5.2.1, 5.2.1[1], [2]	EMC FLARE 03.24, EMC FLARE 03.26		

Figure 1.
This is an example of CLARiiON FLARE and PowerPath compatability. I selected one item, PowerPath 5.2, then browsed the support statement for Windows 2003 and CLARiiON. The navigator is much more powerful than this and can be used to narrow down the entire standard you are trying to set.

Now, back to your standards and how to determine what to include in the standards document. The EMC Support Matrix is referenced on a 6 month cycle to provide a baseline of supported drivers, firmware and applications that are compatible with the OS and hardware. Once this standard is established, a Microsoft standard for SAN attached systems document is published to lock in the revisions so a deployment kit can be created. This matrix is then published to an internal SharePoint site that can be referenced by anyone in the company. The application owners can be confident that the published standard has been tested and can be applied to their environment. They also know that if any problems arise, they can work with the developers of the standard to triage the issue, work out a solution internally, or take the next step to engage vendor support. The standard is what allows the environment to maintain reliability while providing a framework for compatibility across all applications.

From standards to installs

Once the standard is established, all drivers and firmware are downloaded to a central location and rolled into an automation package that can be integrated into Microsoft's deployment process. Testing is performed for multiple scenarios across all currently supported OS versions for both upgrades from the existing standard, soon to be N-1, and new deployments. Issues found during testing are triaged with Microsoft and fixed before the latest versions are released to gold. A test matrix is created for each version of the standard to ensure the installs are compatible with the broad range of Windows versions and applications. In most cases, there are only two currently supported versions of Windows on campus. Two architecture types, x86 and x64, are within those two versions. This is important information for the installation kit because it determines the automation logic for what drivers to install.

Automation

The Sankit automation script is the workhorse of the installation. There is one basic law that the Sankit needs to abide by; *single reboot*. There is a very sound reason for this rule and it involves data protection. When a system is put into maintenance mode, basic steps are taken to ensure that applications are shutdown and the volumes attached to the server are protected. In most cases, services such as SQL and SharePoint are set to manual and shutdown so Sankit can be applied. However, in some rare cases, these safeguards are not taken. This is where the kit works with the safeguards to maintain data integrity.

With a single reboot install, even if SQL is not set to manual when the system comes back online, the drivers will be there so the volumes can be accessed. Data is at risk if the kit required two reboots, one to clean off the old driver, and a second to install the HBA driver or PowerPath. This is due to the OS coming back online without a function driver stack. Corruption could occur before the next phase of installation is concluded if the application tries to access the data through paths that are not filtered or if the HBA driver is an older revision with known bugs.

The entire Sankit is written in a Windows batch commonly known as bat files or cmd commands. This is the native command syntax to a DOS prompt in any windows operating environment. Although this scripting language does not have as many features and functions as vbscript or perl it has one advantage, it is native to the OS. If you would like to use a command that appears when you type help, it will work regardless of which version of Windows you run it on. This is very important to keep the automation simple and reliable.

How does the Sankit work?

Sankit is nothing more than a wrapper that determines what OS type and architecture it is being run on. It calls the vendor specific installs for the HBAs in the machine. As I mentioned earlier, my partners have done a lot of integration work to make this process successful. There are configuration files and command line syntaxes that make things easier and take the burden away from the wrapper and place it on the driver package. I am not trying to undersell what the wrapper does and the complexity of making it work, but it follows the very basic order:

1. What OS
2. What Architecture
3. What HBA type
4. Locate packages that match
5. Install
6. Catch errors
7. Report and Log these errors
8. If no errors ask for a reboot

Let’s discuss how important it is to allow the driver package to handle the logic of the install (item 5). It is simple for new deployments, but much more complex for N-1 upgrades or systems that are even further out of compliance. Consider PowerPath; there are a wide range of PowerPath versions in production at Microsoft. In most cases, we are dealing with an install that is 1 or 2 versions behind, but in some cases we are dealing with even older packages. PowerPath and the team behind the product know the best way to deal with an upgrade and will have integrated this into the installation executable. For this reason, the Sankit does not try to modify or remove the old version first. It calls the install package for the new version and lets the installer handle the logic behind the upgrade.

What about the code?

There has been a lot of talk about automation and standards and even a little theory behind the whole process. In this section, I will offer snips of the kit and explain what the code is doing. These code snippets should get you started on your own standards-based automated installation system for your compute environment.

Verbose output versus simple output.

```
@If "%echoon%"=="" @echo off
```

(Woude, 2009)

This is the very first line in any batch script I create. It allows me to control what someone sees on the screen if they run the script at the command line. The default is only echo out what is explicitly stated. If the user sets the environment variable “echoon” to anything, the script will echo every command that is issued to the screen for more robust debugging.

Set up your environment variables at the top and document in the code.

```
::------------
:: Set local paths
:: and drive letter
::------------
set pt=%~dp0
set drv=%~d0

::------------
:: Return codes
::------------
set rtn=0

::------------
:: Set OS_Type
:: We need to figure
:: out the OS_Type
:: now due to
:: 2008 and 2003
:: support
::------------
set W2003=5.2.3790
set W2008=6.0.6001
set W2008_SP2=6.0.6002
set os_type=

::------------
:: Temp and
:: Version info
::------------
set temp=%systemdrive%\temp\logs
set ver=SAN-Feb_29_2009-v%rev%
if NOT exist %temp% mkdir %temp%
```

The script should document itself. I use the double colon "::" as my comment syntax. REM is used in most batch files but I find it more difficult to read and the "::" is a valid non-error syntax. Set any environment variables at the top and document what each one does. This will be helpful for maintenance and if someone else takes over the code.

Logging

```
::------------
:: Timestamps
::------------
set stamp=%date%_%time%
set stamp=%stamp: =%
set stamp=%stamp:/=:%
set stamp=%stamp::=-%

::------------
:: Log info
::------------
set logdir=%systemdrive%\temp\logs
set logfile=%logdir%\EMC_SanUpdates_%stamp%.log
if NOT exist %logdir% mkdir %logdir%
```

Log as much information as you can. You will only have the information in your logs to debug the problem if and when something goes wrong. I set up my logs based on the timestamp code. You can create a date and time stamp that allows for a unique log for every run of the script by leveraging the set command and the extended feature of character replacement. In the four lines of code the long format of %date%_%time% has spaces, forward slashes, and colons that either can't be used in file names or make the file name difficult to deal with. Each of the three lines replaces these characters with something easier to work with. So the initial variable looks like this:

Sun 02/10/2009_22:57:43.61

And ends up like this:

Sun02-10-2009_22-57-28.85

What OS is this?

```
::-----------------------------
:: Cool thing about this is once we
:: get the os_type we can set
:: the install var based on it
:: and continue on.
:: If OS type does not match
:: quit the script
::-----------------------------
for /f "tokens=1-6 delims=], " %%a in ('ver') do set os_type=%%d
echo return token for ver command is %os_type% >> %logfile%
if "%os_type%"=="%W2003%" set os_type=2003&& goto :check_arch
if "%os_type%"=="%W2008%" set os_type=2008&& goto :check_arch
if "%os_type%"=="%W2008_SP2%" set os_type=2008&& goto :check_arch
echo can not determine OS exit now >> %logfile%
goto :end
```

The ‘for’ loop is the key to handling the majority of difficult tasks in batch scripting. In fact ‘for’, ‘set’, and ‘if’ are the main tools you should understand when writing a batch command. In this example, the ‘for’ loop is parsing the output of the ver command and using ‘set’ to place this data in the environment variable os_type. The three ‘if’ statements check against the known environment variables of the OS version then reset the variable os_type to the Windows major class of 2003 or 2008. This is important because most vendor installation packages have two pieces of data in the name, the Windows version (2003 or 2008) and the architecture (x86 or x64) that we will see in the check_arch function.

What processor architecture is this machine?

```
:check_arch
::----------------------------------------
:: Check the architecture
:: and quit if not correct or assume
:: it will not change while the script is
:: running
:: See ENV var %arch%
::---------------------------------------

if /I "%PROCESSOR_ARCHITECTURE%" == "x86" set arch=x86&& goto :setpacks
if /I "%PROCESSOR_ARCHITECTURE%" == "AMD64" set arch=x64&& set PG_x64=
(x86)&& goto :setpacks
if /I "%PROCESSOR_ARCHITECTURE%" == "IA64" set arch=IA64&& goto :setpacks

echo Processor Check: Package for %PROCESSOR_ARCHITECTURE% not supported
echo Processor Check: Package for %PROCESSOR_ARCHITECTURE% not supported
>> %logfile%
set rtn=5
goto :end
```

Nothing too complex here, there is already a default processor_architecture environment variable in the base install of Windows that contains the type of processor the OS install supports. Convert the AMD64 to the more generic x64. This also sets the “Program Files (x86)” directory up which does not exist on x86 systems.

You can proceed to call the vendor packages now that the logging is enabled, and the OS version and architecture are known. Remember to let them do the heavy lifting and keep in mind that the installation package should not install if it does not find an HBA that it supports. Your wrapper should continue to the next with the assumption that the packages you chose will support the HBAs that your company purchases.

Conclusion

At Microsoft, one of the more challenging problems is how to keep thousands of servers up to date with drivers and firmware for SAN attached hosts in an environment that is spread across many global data centers. These data centers have limited budget, staff are less specialized, and in some cases the data center is completely unstaffed other than security and a few vendors that can power cycle systems. This business is growing, demanding faster deployments with greater uptime while reducing the total number of hands-on activities with each computer. To help Microsoft with these challenges, the EMC account team works directly with IT staff to create a standards-based deployment wrapped in an automation package to allow for rapid and precise provisioning of storage to applications. These standards help with maintaining uptime and application compatibility for production applications.

Works Cited

Pohto, M. (2009). *Green IT in Practice: SQL Server Consolidation in Microsoft IT*. Retrieved from MSDN: http://msdn.microsoft.com/en-us/architecture/dd393309.aspx

Woude, R. v. (2009). *Short Command Line Tips*. Retrieved from Rob van der Woude's Scripting Page: http://www.robvanderwoude.com/

Biography

Aaron Baldie has been working in the IT industry since 1994 where he began his career as an administrator for the University of Washington while getting his undergraduate degree. He has held many positions with technology companies in the greater Seattle area before coming to EMC in 2004 as an Enterprise ATC at Microsoft.

He holds an EMC Technology Foundations (EMCPA) certification.