## Tony Foster's Blog

# Getting Some REST(ful APIs) - Part I

Posted by Tony Foster in Tony Foster's Blog on Nov 14, 2018 4:28:00 PM

This story starts out showing my age relative to the virtualization industry. Many moons ago when I started working on virtualization (13 years ago) everything was a manual process, after all this was ESX 2.0 (no I didn't forget the 'i,' back in those days the hypervisor was a fully functioning OS with a Linux kernel and I could install software on the hypervisor). Everything that was done had to be done manually. REST APIs didn't exist. Only the smart people were thinking about doing something crazy like scripting the deployment and monitoring of virtual environments using automation.

Which is what this post is about. Automating! And more specifically about automating with RESTful (Representational State Transfer) APIs (application programming interfaces). For those not familiar with RESTful APIs, they are an interface to control hardware and software (or really any web service) over HTTP. RESTful APIs send requests to GET, PUT, POST, and DELETE data. The response back is typically formatted as CSV, RSS, HTML, XML, or JSON making the data much easier to work with. I could have done so much more when I started working with ESX had RESTful APIs been available when I started with virtualization.


That's where my story gets going, fresh out of college, starting a new job at a value added reseller (VAR). The CEO of the company I was working for at the time came in to my office and told me he wanted me to figure out how to backup this newfangled software he'd heard about called VMware because it stored entire computer systems as files… and if they were files then they could be backed up, and backup is what the VAR was/is known for!

I set to work researching freezing, thawing, and stunning VMs. Reading anything I could get my hands on to understand what was going on under the covers. Finally I created a bash script that could be called by backup products to enable capturing VMs in ESXi 2.0. It worked spectacularly. The next week ESX 2.5 was announced with all the features my script had built into it.

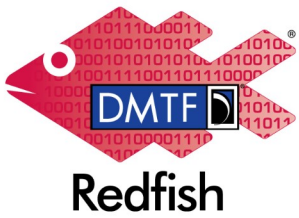
That was a major stepping stone for me and showed the value of being able to communicate with systems outside of the traditional GUI (graphical user interface). As virtualization grew up, I kept learning. Soon I was building scripts to automate the setup of hosts, networking, and storage. All of this was being done through shell commands in putty sessions. You can see one such script I wrote to configure standard vSwitches on my personal blog. Scripts like this made my life easier just not as easy as RESTful APIs do.

Around 2009 to 2010 vendors started to develop customer accessible/documented APIs for their various offerings. This made it easier to programmatically get information from a system (be it hardware or software). One such interface for Microsoft PowerShell, that many are familiar with, is VMware's PowerCLI. It was originally based on SOAP (simple object access protocol) API calls to communicate with components. As time has progressed VMware has introduced REST API calls for many components as well.

I'm not going to spend a lot of time discussing what the SOAP and RESTful APIs are. If you would like to find out more about the differences between RESTful APIs and SOAP APIs, RESTfulAPI.net has a good explanation of the differences here: https://restfulapi.net/soap-vs-rest-apis/.

Suffice it to say, RESTful APIs tend to be more data driven which correlates well with what's happening in data centers these days with AI and machine learning. To feed machine learning what do you need? Data! I even did a workshop at USENIX LISA 16 conference talking about creating intelligent data centers using virtual reality and machine learning. As I have noted for about five years now, data centers generate a plethora of data. RESTful APIs help to make collecting that data possible, making it simpler to manage and automate the components in a data center.

How about an example. Maybe even one from your own data center. Let's look at the five Dell EMC PowerEdge servers you have in your data center. Wait, you have more? So 10 servers? 50 servers? 100 servers? 200 servers? More? Five or ten servers might be easy to manage one by one, but 50 or more becomes a pain. How does one manage at scale? The use of RESTful APIs. The Dell EMC iDRAC with Lifecycle Controller has a RESTful API with the DMTF Redfish standard built in that allows administration and configuration of hosts.



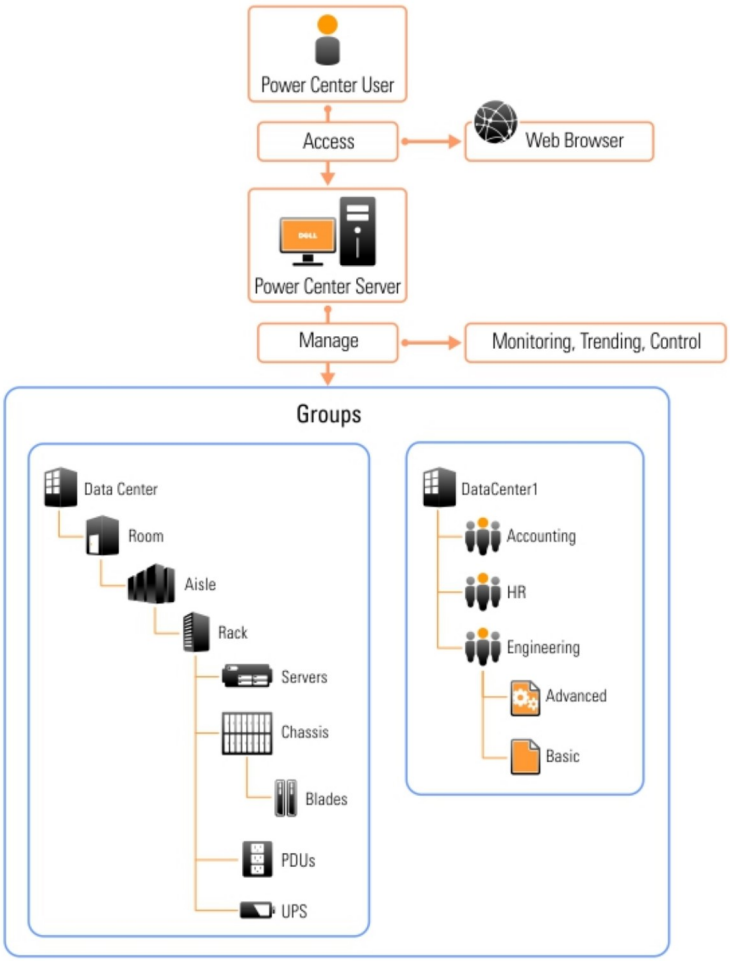

These RESTful APIs make it possible to manage at scale. When ten new PowerEdge servers arrive on the data center loading dock, they can be racked and cabled. Then instead of taking a crash cart to each one and configuring and, if need be, updating the BIOS of each server, you can prepare the server using a Server Configuration Profile (SCP) that is imported from a network share. This of course saves time, allowing IT to become more innovative and spend less time keeping the lights on in the data center.

That's one way RESTful APIs can help. Now, let's expand on this… We could start monitoring the temperature sensors across the PowerEdge servers using the iDRAC Lifecycle Controller and its RESTful API. You've built an interface that is aggregating information from all the temperature sensors using the RESTful APIs. You notice that one

particular server in the middle of the rack keeps getting warmer. (Remember: RESTful APIs tend to be more data driven.)

This allows you to be proactive in the management of the data center. Is it just that server that keeps getting warmer or are all the servers' temps rising? If it is just the one it may indicate that it needs some maintenance, but if it's the center of a temperature spike among many servers, it may indicate a wider problem with data center cooling (bull's eye effect). How cool would it be to go from the micro to the macro level view of the data center with just a few commands? And it's possible to do this using RESTful APIs.

This can easily be expanded beyond just RESTful APIs for PowerEdge servers. Dell EMC has a solution called the Dell OpenManage Power Center (OMPC) that allows you to monitor temperature, power consumption, and much more. The OMPC provides increased visibility over power consumption, anomalies, and utilization through fine-grained instrumentation. This enables increased control, improved rack density, faster response times, greater accuracy, and broader decision-making intelligence than would otherwise be possible in data centers. And as with the temperature spike example from above, it's possible to examine the operating environment at the room, aisle, rack, and component level (see diagram below).




Not only is there a GUI to OMPC, it also has a RESTful API allowing for integration in to existing monitoring systems. This really starts to bring together, or converge, what's happening in the data center, making it easier to keep tabs on those 10, err 20, err 50, err 100, err how many ever PowerEdge servers you have. It also incorporates your server's environment into your existing monitoring and management tools to get the most out of the data center, be that power consumption, workload balancing, CPU consumption, and/or temperature monitoring.

I could keep going on about RESTful APIs with PowerEdge servers, but we've got lots to get to. If you would like to read more about automating PowerEdge server deployments or configuring Redfish, be sure to visit the Redfish knowledge base on the Dell EMC support site here: https://www.dell.com/support/article/us/en/04/sln310624/redfish

I'll continue this RESTful story in my next blog. We'll look at all the other things you can do to improve data gathering and management in your data center. Till next time, may the lights of your data center stay off and your server fans keep humming.

64 Views

Tags: poweredge , openmanage , soap , scp , idrac , rest_api , rest , restful_api , restful , poweredge_servers , openmanage_power_center , ompc , lifecycle_controller , dmtf , redfish

Average User Rating

(1 rating)

My Rating:

0 Comments

There are no comments on this post