Storage Performance Analysis and Alerting Appliance (SPAAA) – Under the Hood

I don’t know when I transformed myself from being an Infrastructure Architect to a Software Architect. But I think the people around me acted as catalyst and injected that mind set, which is a deadly combination, because I see the problems around Virtualization industry.

Since last couple of months, I did not get time to write any blog or not even seen much active in Twitter as well and now you can see what I was Architecting. Please note, I am still not a Software Developer but a Blueprint creator or max you can call me an active guy in Software Prototyping. Now, without any further dilemma, let me tell you the appliance what I was Architecting.

In today’s world, the most strategic and analytic pain point is Storage Virtualization and it’s problems. One of them is doing a Pro Active Analysis and make the vSphere Admin guys understand what could be the corrective action. One of the problems I have factored in, in this Appliance is IO Size from Virtual Machines and what is the correlation of that with RAID Stripe Size. Also another thing which has been factored in, is what would be the %Parallelism based on the RAID Stripe Size.

The IO size is an important characteristic of storage profiles. A variety of best practices have been provided by storage vendors to enable customers to tune their storage to a particular IO size. If the Stripe size is not at least as large as the I/O size then the application IO starts fragmenting and start writing towards the next stripe.
This means your head movement will happen and that means some amount of Seek time and rotational time will be added as penalty per block write.

If stripe size is larger than I/O size by a factor of two or four, then trade-offs may arise. However, user need to understand the basics of performance improvement and then take the call.

The large stripe size can be advantageous because it lets the system perform more sequential operations on each disk; it decreases the number of seeks on disk. Having said that, user need to learn and take a manual call as to what is best and what is not. Lack of user understanding is the biggest challenge and when there is a large vSphere environment understanding the IO Length and dynamically analyze that and suggesting a Stripe Size benefit is more advantageous.

Summary of this appliance and how it can solve the above stated problem

When striping data across drives, you need to be sure that data is being spread across them evenly and that the size of data written to each disk is of optimum size for the type of file you’re working with. The piece of a stripe that’s written to each drive is called a chunk; you can control chunk size in storage subsystem management software.

The RAID chunk size should suit the I/O characteristics of the data you’re working with. The key here is the size of the average I/O request you’re going to place on the RAID array; as a rule of thumb, if you want big I/O requests, you should opt for smaller RAID chunk sizes, and if I/O will be small, you should go for larger chunks.

For example, if your business works with lots of video or large image files, you will want to ensure maximum throughput. That means you will need to spread data across individual drives as much as possible. For this use case smaller RAID chunk sizes (for example, 512 bytes — one block — to 8 KB) fit the bill because you want to take data from one drive while the others seek the next chunks to be read.

At the other extreme in terms of use cases would be running a database in which the amount of data read on each operation is small, say, up to 4 KB. Here you want a single I/O to be dealt with by one drive with one seek action rather than be spilt between more than one drive and multiple seeks. So, for use cases such as databases and email servers, you should go for a bigger RAID chunk size, say, 64 KB or larger.

This is going to be an appliance deployed on top of any vSphere environments for analysis of Storage Performance and Proactive alert generation.

This appliance framework will mainly have four engines in the backend to do all of the permutations and combinations of the various Stripe Sizes of the Array and No. of Spindles in a given Datastore. It will also factor in the I/O length of the Virtual Machines in a given vSphere environments for Storage Performance improvement indicator.

1. Environment onboarding module
2. Gathering I/O Size Module
3. Analyzing I/O Size Module
4. Proactive Alert Module

Environment on-boarding module
****

This module will be used for gathering/on-boarding a vSphere environment and it's related storage(s). That means in the main page an Administrator will be prompted to enter the vCenter details (IP and Admin credential) and it's associated storage details (IP and Admin credential). This will be stored in the database of the Appliance itself.

Gathering I/O Size Module
****

We will not ask user to specify any input for this, rather I/O Size module will automatically analyze the I/O Length of the Virtual Machines using vSCSIStats.

vscsiStats collection and analysis requires two steps:

1. Start statistics collection.
2. View accrued statistics.

The tool will be started with the following command:

/usr/lib/vmware/bin/vscsiStats -s -w <world_group_id>

This command starts the process that will accrue statistics. The world group ID must be set to a running virtual machine. The running VMs' IDs can be obtained by running '/usr/lib/vmware/bin/vscsiStats -l'.

After about 30 minutes vscsiStats will stop running. If the analysis is needed for a longer period, the start command should be repeated above in this window. That will defer the timeout and termination by another 30 minutes. This module will then store the data in the Appliance database.

Now this module will also send a remote call to the storage array using the previously stored data in the database during onboarding. This will fetch the Datastores and it's correlated LUN Stripe Size and No. of Spindles housed there. After getting the data this module will store it in the Appliance Database.

Analyzing I/O Size Module
****

Once the run is over for all virtual machines (using World ID), this module will start analyzing the data in Histogram to understand the I/O Size of each VM. vscsiStats can provide a histogram of I/O sizes to help this process.

Viewing Statistics
++++++++++++++
Counters are displayed by using the following command:

/usr/lib/vmware/bin/vscsiStats -p <histo_type> [-c]

The histogram type is used to specify either all of the statistics or one group of them. Options include all, ioLength, seekDistance, outstandingIOs, latency, interarrival.

Results can be produced in a more compact comma-delimited list by adding the optional "-c" above.

This module will only run "ioLength" to understand the I/O block size of the Virtual Machines..

It will run '/usr/lib/vmware/bin/vscsiStats -p ioLength' command to do that.

It will get the ioLength for each VM and store it in the appliance database. Also this module will run the query on the vCenter Server inventory to fetch the VM's Datastore mapping. These data will be stored in the database table with the following schema:

Table Name: IO_LENGTH_STATS

Fields:

Name: IO_LENGTH
Type: long

Name: Virtual-Machine-Name
Type: String

Name: Datastore-Name
Type: String

Now this module will send a remote call to the Storage Array and then fetch the Stripe Size, No. of Spindles per LUN and it's associated Datastore name and save it in the Database table with the below schema:

Table Name: STORAGE-ARRAY-STATS

Fields:

Name: STRIPE-SIZE
Type: LONG

Name: NO-OF-SPINDLE
Type: INT

Name: DATASTORE-NAME
Type: STRING

Now it will run this query to group the VMs based on the IO Size.

select IO_LENGTH as “IO_SIZE(Bytes)”, (count(*)/sum(count(*))) * 100 as “PERCENTAGE(%)” from IO_LENGTH_STATS group by IO_LENGTH;

Below is a sample output for the above query :-

IO_LENGTH(Bytes)	PERCENTAGE(%)
65536	10
32768	5
16384	10
8192	25
4096	40
2048	5
1024	3
512	2

Proactive Alert Module

****

Now the alerting module will help customer to understand the present situation based on the current Array stripe size and no. of spindles to factor in parallelism. It will also help customers to understand the benefits and penalty in parallelism by changing the Array stripe size.

So, in a nutshell, it will show you the current scenario and the future scenarios (both advantages and disadvantages) by factoring in various combinations. Those combinations are these:

1. I/O Length (Block Size)
2. Stripe Size on Array
3. No. of Spindles

Current situation will be assessed in two magnitude, one will be parallelism and the second one will be Latency indicator based on the following formula:

Parallelism (%) = Ceil (I/O Length / Stripe Size) * 100/No. of Spindle

IO Write Latency Indicator (t) = Ceil (IO Length / (Stripe Size * No. of Spindles))

Where (t) is the time in milliseconds for latency (seek time + rotational time), do note that “t” can vary based on model, vendor, disk type and the position of the head.

These two formulas will also be used to assess the different combinations of Stripe Size, I/O Length and it’s effect on %Parallelism and I/O Write Latency indicator.

Note: This data gathering and analysis can be done in cyclic fashion to provide performance improvements indicators to customers on a repetitive basis over time.

Let me show you an example of the output of what it should look like using some example values of current situation and different combinations.

Current Situation Assessment:
——————————————

Array Stripe Size = 128

I/O Length = 4

No. of Spindle = 4

Parallelism (%) = Ceil (4/128) * 100 / 4 = 25

I/O Write Latency Indicator (t) = Ceil (4/(128 * 4)) = 1 (t)

Combination of attributes assessment:
————————————————

Array Stripe Size = 64

I/O Length = 128

No. of Spindle = 4

Parallelism (%) = Ceil (128/64) * 100 / 4 = 50

I/O Write Latency Indicator (t) = Ceil (4/(128 * 4)) = 1(t)

A full-fledged report will look like this:

VM-Name	I/O Length	Datastore	Array-Stipe-Size	No.-of-Spindles	%Parallelism	Latency-Indicator
VM-A	4	DS-A	128	4	25	1 (t)
VM-B	128	DS-B	64	4	50	1 (t)