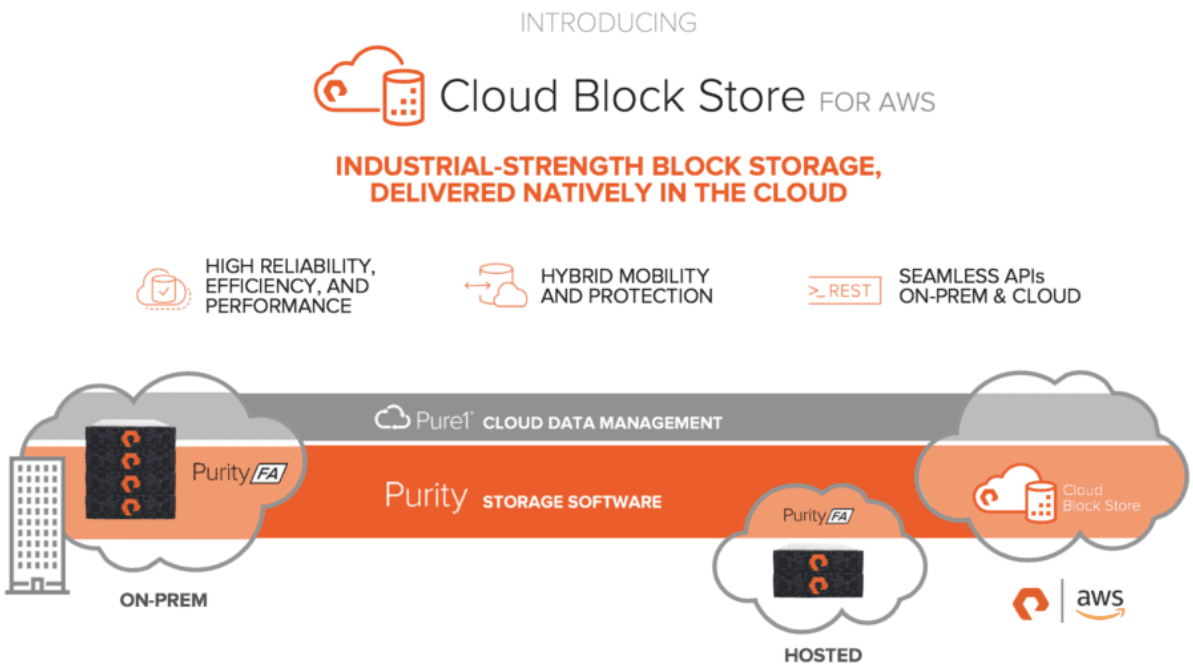
Pure Storage Cloud Block Store for AWS



One of the fundamental features of the operating environment running on the FlashArray™ is the fact that the same software can run on many different hardware implementation of the FlashArray. This is one of the reasons that we can offer hardware Non-Disruptive Upgrades or when we introduce new features (even things as expansive as VVols) we can support it on older hardware. We support VVols going back to the FA 420-an array that was introduced before I joined Pure Storage® 4.5 years ago.

Furthermore, we have been having increasing conversations around the public cloud. Not just running applications in it, but moving data to and from it. DRaaS (Disaster Recovery as a Service) is an increasingly talked about use case. VMware Cloud in AWS is getting more and more attention at VMworld, and in general. We, at Pure get it. Will everything go to public cloud? No. Certainly not. Will everything stay on-premises? Also, of course not. Some customers will. Some will not at all. Many (most?) will use both in some capacity. So enabling data mobility is important.

So how can we at Pure help with data mobility? Well, a couple ways. In Purity 5.1, we introduced Snap to NFS. This allowed us to provide the ability to send our snapshots to NFS targets for retention and moved back to a FlashArray. In 5.2 (coming out quite soon) we offer the ability to send snapshots to S3 targets and then bring them to whatever FlashArray. A lot of potential to continue to build on that too.

That's just part of the story–and certainly not enough.

We recently acquired StorReduce which will provide further benefits as we work to integrate it into our portfolio.

That wasn't enough either, though.

Customers like to protect their data on the FlashArray using our built-in replication. Either our async replication or our synchronous active/active replication. The problem there though is that it requires a FlashArray to be the target. What if they want to replicate into something like AWS?

What if once you get that data into AWS you want to snap it, replicate it back, share it?

I think you can see where I am going.

https://www.purestorage.com/solutions/cloud/multicloud.html

Back to the original paragraph–there is no reason that Purity itself needs to be locked to our hardware. It is custom software that treats the underlying flash specially (whether it is SSDs or our NVMe-based flash modules) but always presents the same thing through our front end.

So here comes what we call Cloud Block Store. Cloud Block Store is the code and features you have come to use and love on your on-premises FlashArray running in an EC2 instance in AWS.

This allows you to use our granular and deep data reduction to deliver capacity and performance economically in AWS. You can present volumes, copy them instantly, snap then, replicate them and run workloads against them to your EC2 instances–or anything with network connection to the VPC in AWS where a Cloud Block Store instance is hosted. Since the front end is thinly provisioned, and we compress and dedupe before we persist it on AWS-provided storage, so Cloud Block Store is always efficient.

The ability to replicate from on-premises FlashArrays to Cloud Block Store (using async or ActiveCluster) provides a way to move data in, and out of AWS. Our data reduction preserving replication reduces the impact of replication throughput on top of that.

You can use the same APIs in AWS as you use on-premises with the FlashArray. Same GUI, same REST, same CLI.

AWS CloudFormation Templates

One of the features AWS offers is CloudFormation templates–this is a method for providing a portfolio of products and those products all have template descriptors on how they are deployed. Very similar to a vAPP content catalog on-premises.

This makes it very easy to deploy Cloud Block Store over and over inside of AWS. Either through the AWS service catalog, or through the AWS API. The AWS PowerShell Module makes this very easy to automate–you could even deploy it through products like vRealize Automation and vRealize Orchestrator–using the vRO plugin for AWS and AWS APIs. Even better–once deployed, Cloud Block Store looks just like a FlashArray to our vRO plugin, so it can be handled and managed with the same workflows you use today in vRO, if you so choose. See a demo of using vRA to deploy it below:

vRA also provides the ability to create and manage EC2 instances–so not only can you manage the Cloud Block Store with our existing vRO plugin, you can use it to provision storage from Cloud Block Store to EC2

instances.

AWS Data Mobility

So one of the things that makes this interesting is sending data sets into AWS. Dev/test or whatever reason you might have. Though if your environment is on VMware how does that work? How do I replicate a VMFS into AWS and use the data in it? The answer is, you don't! VVols my friend.

One of the things I have been chirping about around VVols is the data mobility they offer–well before we even conceived of Cloud Block Store. Each virtual disk is its own volume on the array–a volume that is not formatted or interfered with by VMware in any way–it is simply a block volume with NTFS on it. Or XFS. Or EXT4. Etc.

https://www.codyhosterman.com/2017/12/vvol-data-mobility-virtual-to-physical/

So by putting a VM onto VVols, you can then assign it to our replication using Storage Policy Based Management directly from VMware. And in that policy, make a Cloud Block Store instance the target to replicate to. This will allow you to protect your VM data into AWS.

This is one of the ways VVols is so powerful–it gives you extraordinarily tight integration into your on-premises VMware, but allows you to easily move VVol data volumes to anything else–on-premises, or in this case AWS.

Our VMware strategy has been based on a couple of things:

- VVols–this provides tight integration with VMware and its ecosystem as well as data mobility
- ActiveCluster–this provides the ability to replicate in an active-active configuration, but fundamentally provides the ability to non-disruptively migrate volumes between FlashArrays.
- Purity CloudSnap™ provided ways to move data to non-FlashArray storage
- Pure1® allows the ability to manage it all in one place

Putting these all together is our goal, but it was missing a piece and that is Cloud Block Store. We see great potential in using our replication on-premises to AWS. Or between AWS availability zones. And more. We are still looking into the ways this can be used.

VMware Cloud on AWS is of course of great interest to us–there is no block storage support today (you can do in-guest iSCSI if needed from Cloud Block Store though), but when VMware adds it–we are ready (see their roadmap which lists external storage access publicly here).

We are about to start our limited public beta program, so reach out to your account team if you are interested in trying it out. Or go here:

https://www.purestorage.com/microsites/cloud-storage-beta.html

This has been an especially fun thing for me to work on–I got to learn AWS, its features and API operations (building automation through vRA of AWS) but got to use my existing knowledge of VMware tools and features and of course the FlashArray in it.

This is one of the things I am most excited about–I am very interested in seeing how customers use it. Exploring how this can be handled with VMware Cloud Assembly, managed via lambda requests, AWS RDS interactions–there is a lot to do. So far it has been, well, FUN to explore. Just getting started on this.

As we get closer to GA, watch for more blog posts that are more technical in nature to understand the underlying architecture and what we have done.