Paper SAS5501-2016

Getting There from Here: Lifting Enterprise SAS® to the Amazon Public Cloud

Ethan Merrill and Bryan Harkola, SAS Institute Inc.

ABSTRACT

If your organization already deploys one or more software solutions via Amazon Web Services (AWS), you know the value of the public cloud. AWS provides a scalable public cloud with a global footprint, allowing users access to enterprise software solutions anywhere at any time. Although SAS® began long before AWS was even imagined, many loyal organizations driven by SAS are moving their local SAS analytics into the public AWS cloud, alongside other software hosted by AWS. SAS® Solutions OnDemand has assisted organizations in this transition. In this paper, we describe how we extended our enterprise hosting business to AWS. We describe the open source automation framework from which SAS Solutions OnDemand built our automation stack, which simplified the process of migrating a SAS implementation. We'll provide the technical details of our automation and network footprint, a discussion of the technologies we chose along the way, and a list of lessons learned.

INTRODUCTION

In 2015, SAS Solutions OnDemand began leveraging Amazon Web Services (AWS) to extend its enterprise hosting business to the public cloud. In doing so, we significantly expanded our global hosting footprint, provided a software-defined hosting framework for quicker customer onboarding, and broadened the potential revenue stream of SAS Solutions OnDemand's core hosting business.

The following five fundamental components allowed SAS Solutions OnDemand's entry into the public cloud:

- Teamwork between SAS Solutions OnDemand and the corporate IT teams (SAS IT).
- Creation of a hybrid cloud network.
- Leveraging existing services and policies.
- Automation for the software-defined data center.

Detailed descriptions of these components appear in the ensuing sections.

TEAMWORK

The teamwork between SAS IT and SAS Solutions OnDemand that was responsible for extending SAS into the public cloud is not only a source of pride for the organization, but also an interesting case study of the value of cross-organizational ownership of the components listed previously. There is value in different organizations owning the necessary components of a new framework like SAS Solutions OnDemand's public cloud presence. Even though the software-defined capabilities of a cloud like AWS are transformative (discussed in detail in later sections), it is a mistake to believe that a business unit like SAS Solutions OnDemand can or should own all components by itself.

When SAS Solutions OnDemand first began the nuts-and-bolts discussions regarding how to move into the public cloud, we objectively considered overhauling our entire hosting approach. This discussion included abandoning IT leveraged services, such as authentication via Active Directory, Puppet automation, and compliance processes that depended on existing and hardened on-premises hosting policies. This was the right conversation to have because carrying on-premises technical debt to the public cloud can slow delivery and hinder the adoption of new cloud technologies.

Through these discussions, it was quickly determined that abandoning all of these time-tested and valuable services would be foolish; were we to do so, we would have to spend months or years recreating a cloud-native framework that would allow us to host our enterprise customers in a secure, compliant, and automated public cloud environment.

Once we determined to leverage existing SAS IT hosting services, the question became *Who owns what?* Determining the answer to this took time, but in the end we arrived at the following breakdown:

- SAS IT owns and supports the following components:
  - The hybrid network between SAS corporate and AWS.
  - Operating system (OS) level aspects of deployed hosted systems, including OS maintenance and authentication.
  - Applicable leveraged services (Active Directory (AD), Domain Name Server (DNS), File Transfer Protocol (FTP), and so on).
- SAS Solutions OnDemand owns and supports the following components:
  - Deployment of customer environments.
  - Monitoring and alerting.
  - Customer security (that is, security groups) and AWS console / application programming interface (API) access and policies.

This separation of ownership provides SAS Solutions OnDemand the agility it needs to deploy customer environments quickly and automatically, while relying on SAS IT's robust knowledge and hardened services to support the network and OS. The real magic that was accomplished with these efforts is the resulting ability to deploy production customer environments to an entirely new data center (AWS) and the resulting servers (instances) are managed, monitored, and authenticated exactly as if they were on-premises in the SAS corporate data center. The teamwork and ongoing relationship between SAS Solutions OnDemand and SAS IT that provided these results represents a tremendous victory for SAS.

HYBRID CLOUD

Hybrid Cloud has been one of the most discussed topics in IT since the mid-2000s. In fact, poking fun at trying to define *hybrid cloud* has become about as common as actually trying to define it. For the purposes of this paper, we ignore the broader definitions and focus on what *hybrid cloud* means for SAS Solutions OnDemand's public cloud presence and the value it has provided.

By leveraging the AWS Direct Connect and hardware virtual private network services, we are able to connect the SAS corporate network with the AWS network. Specifically, this allows us to create AWS Virtual Private Clouds (VPCs) that are reachable from our corporate network. This is the only way we could have accomplished the reuse of leveraged hosting services discussed earlier. Figure 1 contains a high-level diagram of how this works for SAS.

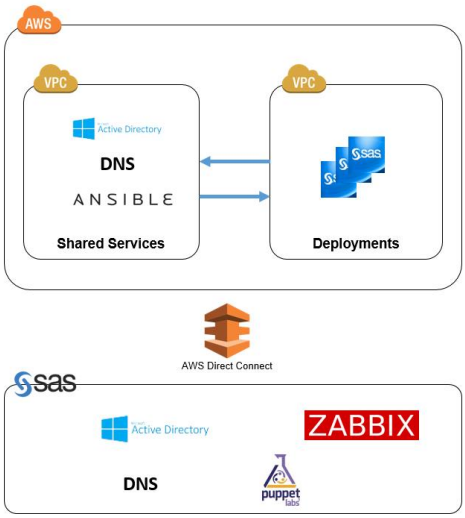

Figure 1: High-Level Hybrid Network Diagram

Many of the lower-level details, including subnets, security group rules, Network Access Control Lists (NACLs), and peering rules, are not included here, to simplify the picture for purposes of this topic.

In AWS, we use one VPC, called Shared Services, which provides the services we leverage from SAS IT. Specifically, we have promoted AD servers and DNS servers. The reason to put these services into AWS is to prevent latency for the SAS instances that require them. Services having to go all the way back to the SAS on-premises network to resolve DNS or authenticate users would result in excessive latency, and this is especially true as we roll our hybrid footprint out to AWS regions around the globe.

The Deployments VPC is where we deploy hosted SAS solutions. It is peered with the Shared Services VPC so that deployed solutions can leverage the shared services. Specific security group rules are in place to allow only the access needed across the peered connection. This is also true for the hybrid network provided via AWS Direct Connect – specific, least-privilege SAS firewall and AWS security group rules are in place to allow only the traffic needed across the network boundaries.

In the end, the SAS-corporate-to-AWS hybrid cloud is basically just a hybrid network, without which we would have been unable to deploy our SAS Solutions OnDemand hosted customers to the public cloud in 2015.

LEVERAGED SERVICES

Now that we have discussed the general value and basic needs of extending SAS Solutions OnDemand's hosting business to the public cloud, we look more closely at the specific services we carried with us. Through the lifetime of SAS Solutions OnDemand, we have steadily enhanced the services that surround its enterprise hosting business. These services comprise a framework that provides automation, security, uniformity, and compliance.

Definitions for these components appear below, and convey why each is so valuable to the framework:

- Automation –Ability to quickly create and configure hosted customer servers, so that they are ready for SAS installations, and the ongoing ability to manage and monitor these servers.
- Security – Segregation of customer environments at the network level and insurance that customer data is protected and isolated.
- Uniformity – Assurance that, along with automation, the base servers are the same and can be accessed in the same way by SAS Solutions OnDemand SAS Administrators and customers.
- Compliance –Controls and processes around the other three framework components, allowing us to host enterprise customers and their sensitive data.

Given these definitions, Table 1 contains the leveraged services that SAS extends to its SAS Solutions OnDemand public cloud environment, as well as the role(s) that each service plays in the framework.

Service	Role(s)
Active Directory	Security, Uniformity
DNS	Automation, Uniformity
Puppet	Automation, Compliance, Uniformity
Zabbix	Compliance, Uniformity

Table 1: Services and Roles

Leveraging these existing systems provides us with an automated and secure hosting environment and also satisfies compliance needs, without having to reinvent the wheel.

As described in the preceding sections, we are beginning to stack frameworks, which allow us to achieve our ultimate goal of extending SAS Solutions OnDemand's business to the public cloud and realize the global reach and increased revenue stream that comes with that. So far, we have stacked leveraged services and the resulting compliance on top of the basic infrastructure and network layers that give us the foundation. The final piece is the set of new automation tools and techniques needed to deploy to a software-defined data center like AWS.

AUTOMATION AND NETWORK

After we decided on our general approach, previously outlined, we needed a mechanism to deploy the base set of resources in AWS that comprise a hosted customer environment. Specifically, we needed automation that can deploy AWS EC2 instances, storage, security groups, and Elastic Load Balancers (ELBs). Without having automation that we could leverage from our on-premises data center, we set out to determine the best technologies and techniques to use. Our initial decision was Ansible. Not only does Ansible have a robust set of features, called *plays*, that provide automated server configuration, it also has a broad set of plays for interacting with various public cloud APIs to create cloud resources. The idea is that one uses the cloud plays to launch cloud resources, then transfer these resources to the configuration plays to configure their operating systems. Our first version of public cloud automation used Ansible exclusively.

After our initial test deployments, we discovered the following drawbacks to the Ansible-only approach:

- While Ansible's cloud plays are valuable, it can be awkward when there is no play for a particular cloud's API (for example, tagging EC2 storage volumes upon creation).
- There is no built-in way to roll back all deployed resources when deployment errors are encountered.
- Creating a SAS Solutions OnDemand customer deployment template using Ansible's playbook paradigm proved challenging, because customer environments vary significantly with respect to the number of servers and storage needs.
- Managing the Ansible inventory of customer instances via the open source EC2 inventory file proved slow as the number of instances increased.

Our conclusion was that Ansible is a fantastic tool for wrapping SSH commands, and configuring and managing a large set of customer servers, but it is not ideal for launching those servers and corresponding cloud resources. These discoveries led to the decision to evolve our automation tooling, as follows:

- Use cloud-native APIs and services for launching cloud resources.
- Use Ansible programmatically for system configuration.
- Establish a clean, programmatic handoff between the cloud-native launch of resources and the Ansible configuration of those resources.
- Establish a final programmatic handoff to SAS IT services for final automation and ongoing management.
- Implement this new framework as a set of microservices with a central API, so that it can be integrated into other business processes later.

We completed this system and have used it successfully to deploy enterprise hosted customers to AWS. The first step involves an internal SAS administrator interacting with our web application to specify details of a new customer environment, such as the number of servers, the file system names and sizes for each server, and the number of Windows terminal servers needed for the environment. Alternatively, the same specification can be passed directly into our REST API by other clients as a JSON document.

After we submit the deployment and it enters Phase One of our automation, we transform the simple JSON into an AWS Cloud Formation document that we then upload and run using the AWS API. This step creates all of the server resources (instances, storage, security groups, and so on). Using Cloud Formation to accomplish this gives us not only all the power of the full AWS API, but also a tremendously convenient way of rolling the deployment back if we encounter errors. By simply deleting the Cloud Formation stack, all of the resources are removed.

After Phase One completes, we enter Phase Two, which leverages Ansible to mount the file systems, make API calls to SAS IT to set up the new server as a monitored resource, and install any base OS packages that are needed before the programmatic handoff to SAS IT's Puppet. The key to this step is

the ability to drive Ansible programmatically, which solves our previous problem of not being able to create Ansible playbook file templates well enough to support our varied customer environments.

At this point, the new AWS instances are deployed, configured to use the leveraged services described previously, and can be accessed by SAS installers in the same way they access servers deployed in the SAS Cary data center.

CONSIDERATIONS FOR RUNNING SAS IN AWS

When considering whether to run SAS in AWS, the following critical questions should be asked:

- What types of workloads will I run?
- What are my storage and compute requirements?
- What are my system recovery requirements?
- What are my security and encryption requirements?

Knowing the type of workload you wish to run is critical, because it drives decisions regarding the level of performance needed, which further defines the storage and compute capacity requirements. Specifically, it is important to determine how intensively your workloads will use disk I/O, because this relates to the available disk (storage) technologies available in AWS (described in detail below).

Decisions on storage and compute needs are connected in AWS. There are two types of storage to consider:

- Instance storage (also called ephemeral storage).
- EBS storage, which is network attached.

Each storage type has advantages and disadvantages. Instance storage is typically much faster than EBS, but is reset (resulting in potential data loss) if the instance is restarted in the AWS console. For these reasons, the `saswork` file system is well-suited for instance storage. Because EBS storage is network attached, it incurs performance penalties, but is much more resilient and also has powerful and convenient snapshotting utilities. EBS is also available in two flavors: standard and provisioned IOPS. For certain scenarios, purchasing provisioned IOPS EBS volumes can enhance performance. Additionally, you can achieve enhanced performance by implementing a RAID0 configuration for either instance or EBS storage.

Options for SAS file systems in AWS (excluding magnetic storage) include the following:

- Instance storage.
- EBS standard.
- EBS Provisioned IOPS.
- RAID0 of all of the above.

Note: Using RAID10 for mirroring is not recommended for performance reasons because it incurs a significant write time penalty. A RAID0 configuration can dramatically increase performance because it achieves the sum of the throughput of each EBS volume. The downside is that it is impractical to use EBS snapshotting to back up RAIDed volumes and other solutions must be used for backup activities. For a small number of production environments, we recommend using a simple, automated copy process, such as a nightly cron-based `rsync` to copy data from the RAIDed filesystem to an EBS volume, which can then be snapshotted to backup data. For large scale of deployments, we recommend pursuing one of the many commercially available backup and recovery tools.

The final wrinkle in storage considerations relates to an instance's network connection to its EBS storage. Different AWS instance types provide different storage connection speeds: some offer 1 Gigabit/second, and others offer 10 Gigabit/second. It is critical to understand what the storage *ceilings* are and where they hit each other.

Consider a test executed with an AWS r3.2xlarge instance, which has 8 vCPUs (the equivalent of 4 cores, when sizing for SAS software). We ran our performance test tool (iotest.sh) with two EBS-provisioned IOPS volumes in a RAID0 configuration. We chose 1280-provisioned IOPS for each volume, because we used the recommended SAS block size of 256KB and, according to AWS, there is little to no value of higher-provisioned IOPS when using this block size.

In the test scenario described above, we should achieve a total throughput of 640 MB/s (megabytes per second) which equates to 160 MB/s per core because, by virtue of the RAID0 configuration, we achieve the sum throughput of the two EBS volumes, each of which provides 320 MB/s. Table 2 shows the actual throughput observed during the test:

Average Read Time (s)	Average Read Throughput Rate (MB/s)	Aggregate Read Throughput Rate (MB/s)	Average Write Time (s)	Average Write Throughput Rate (MB/s)	Aggregate Write Throughput Rate (MB/s)
2405.63	28.94	115.76	2217.19	31.4	125.6

Table 2: Test Throughputs

Notice that the read and write throughput are closer to 30 MB/s per core than 160 MB/s. This is because the throughput of the EBS volume configuration is hitting the ceiling of the 1 Gigabit network connection provided by the r3.2xlarge type to its storage. Simple math yields the per-core theoretical limit of the r3.2xlarge instance type:

(1000 bits) / (8 bits/byte) / 4 cores = 31.25 MB/s per core

This is in line with the observed throughput from our test. We take this one step further and derive the expected per-core throughput for instance types that offer a 10 Gigabit EBS network connection. We'll use the c3.8xlarge as an example:

32 vCPUs (16 cores): (10000 bits) / (8 bits/byte) / 16 cores = 78.125 MB/s

important takeaway message is that any I/O-intensive SAS workloads that demand high-performing storage should be deployed using AWS instances that provide a 10 Gigabit storage network. This limits the choices of instance types, but ensures a better performing environment for I/O-bound workloads.

Finally, it is important to consider your data security and encryption needs for running SAS in AWS. AWS offers EBS encryption, which provides whole disk encryption only. We recommended that you use this by default, but that you supplement it by using SAS Secure encryption wherever available. This provides you with data center-level encryption via EBS encrypted volumes and encryption at rest for your data within the EBS volumes.

CONCLUSION

One can certainly achieve quick success with SAS solutions in a public cloud like AWS, but moving or newly deploying an enterprise SAS environment to AWS requires much more forethought and consideration. In this paper we addressed these and provided a blueprint of how deploying and hosting an enterprise SAS environment in AWS can be done successfully. Our general recommendations are the following:

- Know your workload.
- Leverage existing IT services and processes, where appropriate.
- Leverage as many software-defined cloud capabilities as possible, for automation.
- Carefully consider your storage and compute needs when designing your solution.

By following this basic blueprint your chances of successfully hosting a high performing, compliant enterprise SAS environment in the cloud are greatly increased.

RECOMMENDED READING

- *Performance and Tuning Considerations for SAS® Grid® Manager 9.4 on Amazon (AWS)Cloud using*

Intel Lustre File System, https://support.sas.com/rnd/scalability/grid/SGMonAWS.pdf

- *AWS Direct Connect, https://aws.amazon.com/directconnect/*
- *Amazon Virtual Private Cloud (VPC), https://aws.amazon.com/vpc/*
- *Amazon EBS Volume Types, http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html*
- *Amazon EBS Volume Performance on Linux Instances, http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSPerformance.html*
- *RAID Configuration on Linux, http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/raid-config.html*
- Usage Note 51660: Testing Throughput for your SAS® 9 File Systems: UNIX and Linux platforms, *http://support.sas.com/kb/51/660.html*
- *When the Answer to Public or Private? Is Both: Managing a Hybrid Cloud Environment*, Paper SAS5501-2016

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Ethan Merrill
SAS Institute, Inc.
+1 (919) 531-2241
Ethan.Merrill@sas.com
http://www.sas.com

Bryan Harkola
SAS Institute, Inc.
+1 (919) 531-5604
Bryan.Harkola@sas.com
http://www.sas.com