International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 9 Issue 3 ǁ 2021 ǁ PP. 71-75

Implementation of STS agent in Hybrid Cloud Infrastructure

Dheeraj Parwani[1], Ashish Tiwari[2] Hemendra Khedekar[3]
*[*1]Department of Computer Science Engineering, Vindhya Institute of Technology & Science, Indore, India*
*[*2]Department of Computer Science Engineering, Vindhya Institute of Technology & Science, Indore, India*
*[*3]Department of Electrical Engineering, Swami Vivekanand College of Engineering, Indore, India*
Corresponding Author: Hemendra Khedekar

Abstract

In Cloud Computing, Session token service (STS) enables to integrate directory services of organization with cloud. The employee of an organization is authenticated with his credentials by organization LDAP server, once authorized a temporary token is generated that allows user to login to the cloud account. This is token that is valid temporary and expires after due timeout. The User can login to cloud account and can assume role based on the privileges and policies he is permitted for. Once authenticated by LDAP and user is provided roles he is allowed for. Once the user is authorized, he is issued token to login to cloud account and access all the permitted resources and perform all permitted action and jobs.

In this thesis I am proposing a machine agent to take care of this at the server level and that by passes the session key and access key to store for each account which is not good from security perspective and is troublesome to manage for multiple accounts. If incorrect session keys are used it may result in wrong doing into incorrect account.

Keywords: *Cloud Computing, Identity and Access Management (IAM), Security Token Service (STS), Command Line Interface (CLI), Hybrid Cloud.*

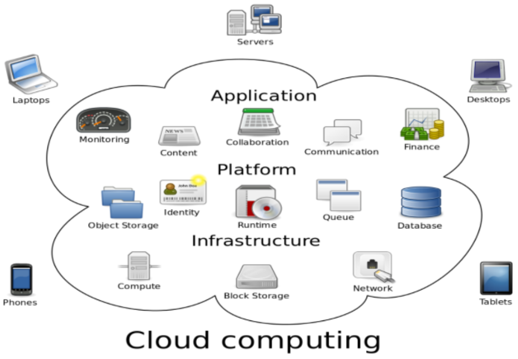
Date of Submission: 09-03-2021 Date of acceptance: 23-03-2021

I. INTRODUCTION

1.1 Cloud Computing

The on-demand delivery of Information Technology (IT) resources over cyberspace with pay-as-you-go pricing is called Cloud Computing. Despite buying, owning, and maintaining physical data centres and servers, we can access technology services like computing power, storage, and databases, whenever required from a cloud provider like Amazon Web Services (AWS).

Simply put, it refers to offering computing services — including servers, storage, databases, networking, software, analytics, and intelligence — over the Internet ("the cloud") to offer faster innovation, flexible resources, and economies of scale. You are required to pay only for cloud services you use, which helps to lower your operating costs, run your infrastructure efficaciously and scale as your business needs change.



1.2 Who uses Cloud computing?

Today, any industry or organization of varying size requires the cloud for a wide variety of cases, such as data backup, disaster recovery, emailing, virtual desktops, software development and testing, big data analytics, and customer-facing web applications.

For instance, a healthcare company may use the cloud to develop more personalized treatments for its patients. A banking or finance company shall use the cloud to power its real-time fraud detection and prevention. The cloud has proven itself to be of benefit to gaming companies as they use its cloud to deliver games online games to players around the world.

1.3 Advantages of Cloud computing

Cloud computing is a big shift that involves switching from the traditional way businesses think about IT resources. Following are the six common reasons organizations are turning to cloud computing services.

1.3.1 Ability

The cloud offers you easy access to a wide range of technologies so that you can innovate more often and build almost anything that you can imagine. You can quickly spin up resources when you need them ranging from infrastructure services such as compute, storage, and databases, to the Internet of Things (IoT), machine learning, data lakes, analytics, etc.

You can deploy technology services in no time and get from idea to implementation several orders of magnitude faster than ever before, which lend you the opportunity to experiment, provide a different experience to customers with new ideas, and helps to transform business.

The largest cloud computing services work on a worldwide network of secure data centres, which are constantly upgraded to the latest versions of faster and more efficient computing hardware. This extends several benefits over a single corporate data centre, including reduced network latency for applications and greater economies of scale.

1.3.2 Elasticity

Now with cloud computing, you reduce bazillions of resources upfront to handle peak levels of business activity in the future. Rather, you provision the number of resources that are required. You can scale these resources up or down instantly to grow and shrink capacity as your business needs change.

The advantages of cloud computing services include the ability to scale elastically. In terms of cloud speak that means delivering the right amount of IT resources required, for instance, more or less computing power, storage, bandwidth, right when it is needed and from the right geographic location.

1.3.3 Cost Saving

The cloud also allows you to trade capital expenses, like data centres and physical servers, for variable expenses and only pay for IT as you consume it. Additionally, the variable expenses significantly decrease than what you would pay to do it yourself because of the economies of scale.

IT experts required to manage infrastructure, buying hardware and software, doing set up, running on-site datacentres – organized serves, the round-the-clock power consumption of power and cooling all the parameters cost gets reduced with the help of cloud computing.

1.3.4 Deploy Globally in No Time

With the cloud, you may expand to new geographic regions and deploy globally in no time. For example, AWS has infrastructure all around the world, so that you can deploy your application at several physical locations with just a few clicks. Putting applications close to end-users results in declining latency and improves their experience.

1.3.5 Productivity

On-site data centres largely require a lot of “racking and stacking” that involves hardware setup, software patching, and various other time-consuming IT management chores. As a result, IT teams now spend more time on achieving more important business goals.

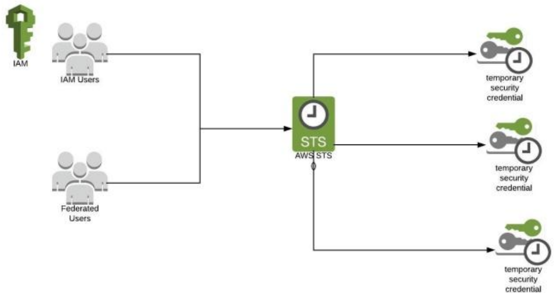
1.3.6 Reliability

Since data gets imitated at multiple sites on the cloud provider’s network, data backup, disaster recovery becomes easy and less expansive which directly helps in business continuity. Various cloud providers offer a wider set of policies, technologies, and controls that not only strengthen your security posture overall but also help to protect your data, apps, and infrastructure from potential threats.

II. PROPOSED METHOD

2.1 System Overview

AWS Security Token Service (STS) that enables you to request temporary, limited privilege credentials for IAM Users or Federated Users).


IAM
IAM Users
Federated Users
STS
AWS STS
temporary security credential
temporary security credential
temporary security credential

Use Cases

Identity Federation (Enterprise Identity Federation [Active Directory/ADFS]/ Web Identity Federation (Google, Facebook)

Cross-account access (For Organization with multiple AWS accounts)

Applications on Amazon EC2 Instances

***S*tep1**

Create an IAM user

Go to AWS Console → Security, Identity, & Compliance → IAM → Users → Add user

Step2

Create Roles

Choose Another AWS account

Step3

Update/Modify Trust Relationships

Go to the Role we have just created and Click on Second Tab Trust relationships

Step4

Add inline policy to the user we have created

Step5

Testing

In this research I am creating a machine agent that will be installed on the machine inside the organisation where user will login with his/her organisational username and password once authenticated agent will give option to choose account and role in the accounts, he has access to. Once user selects account and role, he want himself/herself to assume to, he will be granted access and he will be able to perform all action that he has permission to the account selected.

III. IMPLEMENTATION

3.1 Implementation Détails

3.1.1 Software Requirement

Operating System (Windows or Linux): Our STS script can support in various flavours of operating system have required prerequisite.

This agent has been tested on RHEL5, RHEL6, RHEL7, AWS Linux, AWSLinux2, Windows10, Windows Server 2012, Windows Server 2016, Ubuntu18.04, SUSE15, Debian Stretch and Debian Jessie. This should however work on any system that has Python 2.7 installed and can run the AWS CLI

3.2 Easy & Efficient Project Management

Keeping an obvious overview regarding large software, with many folders as well as files, and millions of lines regarding code, can be a daunting process. Net Beans IDE delivers different views of one's data, from many project glass windows to helpful tools for putting together your software and handling them successfully, letting a person drill into your data efficiently, while providing you version instruments via Subversion, Mercurial, and Git integration out from the box.

Python :- Python is a great object-oriented, interpreted, and interactive programming language

AWS CLI :- The AWS Command Line Interface (CLI) is a unified tool to manage your AWS services.

Linux shells – Use common shell programs such as bash, zsh, and tcsh to run commands in Linux or macOS.

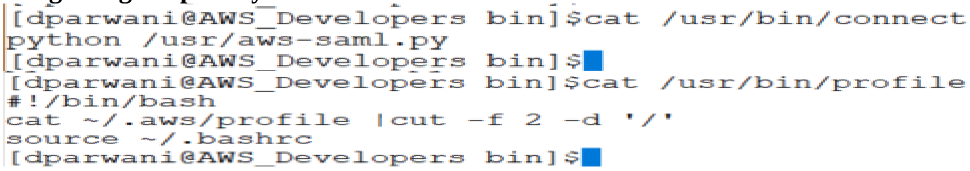
Windows command line – On Windows, run commands at the Windows command prompt or in PowerShell.

Remotely – Run commands on Amazon Elastic Compute Cloud (Amazon EC2) instances through a remote terminal program such as PuTTY or SSH, or with AWS Systems Manager.

IV. RESULT ANALYSIS

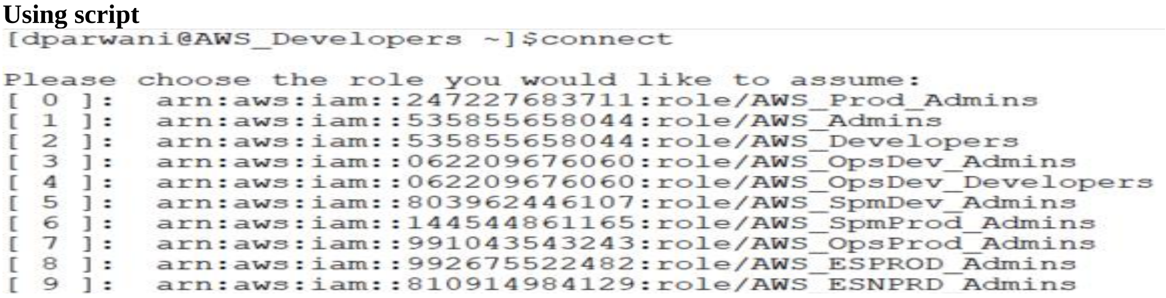
This chapter demonstrates the results and screenshots we have captured during the execution of our STS python scripts which makes call STS login URL and fetch results pertaining to user credentials and give him option to login to one of authorised roles in all AWS account.

Integrating script as system command



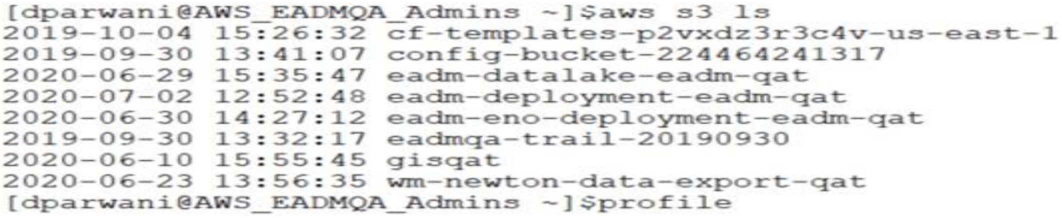
```
[dparwani@AWS_Developers bin]$cat /usr/bin/connect
python /usr/aws-saml.py
[dparwani@AWS_Developers bin]$
[dparwani@AWS_Developers bin]$cat /usr/bin/profile
#!/bin/bash
cat ~/.aws/profile |cut -f 2 -d '/'
source ~/.bashrc
[dparwani@AWS_Developers bin]$
```

Using script



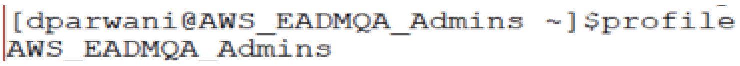
```
[dparwani@AWS_Developers ~]$connect

Please choose the role you would like to assume:
[ 0 ]:   arn:aws:iam::247227683711:role/AWS_Prod_Admins
[ 1 ]:   arn:aws:iam::535855658044:role/AWS_Admins
[ 2 ]:   arn:aws:iam::535855658044:role/AWS_Developers
[ 3 ]:   arn:aws:iam::062209676060:role/AWS_OpsDev_Admins
[ 4 ]:   arn:aws:iam::062209676060:role/AWS_OpsDev_Developers
[ 5 ]:   arn:aws:iam::803962446107:role/AWS_SpmDev_Admins
[ 6 ]:   arn:aws:iam::144544861165:role/AWS_SpmProd_Admins
[ 7 ]:   arn:aws:iam::991043543243:role/AWS_OpsProd_Admins
[ 8 ]:   arn:aws:iam::992675522482:role/AWS_ESPROD_Admins
[ 9 ]:   arn:aws:iam::810914984129:role/AWS_ESNPRD_Admins
```

User Authenticated – Access Role



```
[dparwani@AWS_EADMQA_Admins ~]$aws s3 ls
2019-10-04 15:26:32 cf-templates-p2vxdz3r3c4v-us-east-1
2019-09-30 13:41:07 config-bucket-224464241317
2020-06-29 15:35:47 eadm-datalake-eadm-qat
2020-07-02 12:52:48 eadm-deployment-eadm-qat
2020-06-30 14:27:12 eadm-eno-deployment-eadm-qat
2019-09-30 13:32:17 eadmqa-trail-20190930
2020-06-10 15:55:45 gisqat
2020-06-23 13:56:35 wm-newton-data-export-qat
[dparwani@AWS_EADMQA_Admins ~]$profile
```

User Authenticated – Check Role



```
[dparwani@AWS_EADMQA_Admins ~]$profile
AWS_EADMQA_Admins
```

V. CONCLUSION & FUTURE WORK

Using our STS python script, we have demonstrated how a user can switch between various accounts and roles seamlessly without memorising or keeping track of secret keys and have always fear of keys being compromised. Thus the goal of this research was to show by taking advantage STS and SAML authentication we can handle authentication and authorisation mechanism into cloud infrastructure efficiently. This chapter summarizes the work done in this thesis and then the future scope is given.

In this research work, we created a machine agent that will be installed on the machine inside the organisation where user will login with his/her organisational username and password once authenticated agent will give option to choose account and role in the accounts, he has access to. Once user selects account and role, he wants himself/herself to assume to, he will be granted access and he will be able to perform all action that he has permission to the account selected.

Using our STS python script, we have demonstrated how a user can switch between various accounts and roles seamlessly without memorising or keeping track of secret keys and have always fear of keys being compromised. Thus, the goal of this research was to show by taking advantage STS and SAML authentication we can handle authentication and authorisation mechanism into cloud infrastructure efficiently. Thus, it saves much time and considered as an efficient method as proved from the results.

REFERENCES

[1]. A Complete Guide For Beginners to Learn Amazon Web Services - Xavier Baker

[2]. AWS Cloud for Project Managers - Zareef Ahmed

[3]. AWS Amazon Web Services 2020, Step by Step Guide to Master AWS - Ezra Sabet

[4]. Guide to Learn Cloud Computing & Big Data Storage - Joseph K. Brown

[5]. Overview of Amazon Web Services - Amazon Web Services, AWS Whitepapers

[6]. AWS Certified Cloud Practitioner Study Guide - Ben Piper, David Clinton

[7]. An Overview of the AWS Cloud Adoption Framework - AWS Whitepapers

[8]. AWS Certified SysOps Administrator - Sam Alapati

[9]. AWS Command Line Interface - Jerry N. P.

[10]. AWS IAM Third Edition - by Gerardus Blokdyk
[11]. Mastering AWS Security - by Albert Anthony
[12]. Overview of access management: Permissions and policies – AWS
[13]. AWS Identity and Access Management User Guide: AWS
[14]. IAM User Guide Hardcover – by Documentation Team
[15]. AWS IAM Third Edition Paperback – by Gerardus Blokdyk
[16]. Welcome to the AWS Security Token Service API Reference – AWS
[17]. AWS Security Cookbook - By Heartin Kanikathottu