Penetration Testing - AWS S3, EC2

Pranamya Akella
New York Institute of Technology
Vancouver, Canada
pakella@nyit.edu

Saranya Govvala
New York Institute of Technology
Vancouver, Canada
sgovvala@nyit.edu

***Abstract*— Within the last few years cloud has become one of the major platforms for storing, retrieving and managing business or any type of data. There are many companies today acting as cloud providers by offering various cloud services for the end users. The common aspects when dealing with such data in the cloud is availability and the cost. The cloud gave us the confidence through its backup servers and other domains to achieve availability and with its pay as you use policy the cost is never an issue in the cloud. However, one thing that needs to be considered as critical is security, as security can never be fully achieved and there would always be security issues whenever data is hosted or stored in the internet. In order to assess the security of S3 buckets and EC2 instances, we need a strong testing mechanism that scans, analyzes and exploits existing vulnerabilities. In this project, we are going to perform penetration testing on the above mentioned AWS services in order to assess its strength as well as vulnerabilities.**

Keywords*—*penetration testing, S3, bucket, object, EC2 instances, kali linux, IAM, Jenkins

I. Introduction

The major asset of any individual or an organization is data and moving data into the cloud requires a greater security assurance as cloud means being on the internet and one simple mistake can compromise the Confidentiality, Integrity and Availability of the data. With the information being available on the internet, there is always a chance for an Intruder to break the security and access the confidential data. In the cloud environment, the most common cyber security attacks that have occurred in the recent years were all on improperly misconfigured storage buckets. Also, the latest reports show that exploiting vulnerable EC2 instances that have public access has become the base reason for major attacks. And, installing vulnerable applications on EC2 VMs could also lead to the exploit of EC2 instances. For example, installing a vulnerable Jenkins server by configuring it with default admin credentials could lead to the attacker gaining unauthorized access to the script console by using tools such as metasploit.

II. Problem statement

Amazon Web Services (AWS) is currently the world's most adopted cloud platform that offers on-demand cloud computing platforms by providing various products that include compute, networking, storage, database, IoT, mobile, security etc. AWS offers various services such as Amazon GuardDuty, AWS Inspector, EC2, CloudHSM, EBS, S3 etc., on a pay-as-you-go basis. Out of all, the AWS S3 buckets and EC2 instances are one of the popular attack surfaces as they are most prone to hacking and similar attacks. This project mainly focuses on identifying vulnerable or poorly configured S3 buckets and exploiting vulnerable EC2 instances by installing vulnerable applications. We can exploit such buckets by extracting sensitive information and by injecting malicious code. One popular method to exploit EC2 instances is by gaining script console access by installing vulnerable applications such as Jenkins, to run malicious commands.

III. Review of related work

In the past, researchers have published several articles on how vulnerable S3 buckets and EC2 instances are:

S3 buckets - Usually, bucket permissions can be applied either to the bucket or to the objects inside the bucket. However, if the bucket permissions are set to be public, then anyone can access the objects inside the bucket even though the object permissions are declared as private. This common S3 vulnerability has been exploited in many ways to gain unauthorized access to the bucket contents. However, in our project, we are launching an XSS attack by uploading malicious JavaScript file into the bucket to exploit any web application that tries to access the vulnerable bucket.

EC2 instances - As EC2 instances are connected using RDP or SSH sessions from hosts outside the network, they are usually configured with default access control options such as configuring source IP addresses as 0.0.0.0/0 which allows any machine to connect to our instances. This is a very common vulnerability. Another common vulnerability is not encrypting the contents of EC2 instances, which leads the attacker to access all the confidential data in simple plain text. When a network or security group is assigned with 0.0.0.0/0 then it means that anyone in the entire world can access the devices in our AWS environment through SSH and RDP. This is a serious problem in situations where an EC2 instance consisting of a misconfigured application can be easily found with the help of various tools and can easily be exploited, compromising the CIA of the system.

IV. Proposed solution

In this project, our major focus is to perform penetration testing on AWS S3 buckets and EC2 instances. We are going to implement all 6 phases of penetration testing which are Information gathering, planning, vulnerability scanning, exploitation, post-exploitation and reporting. After successfully launching the attack we have to analyze the

strength and vulnerability of s3 buckets as well as EC2 instances. One thing that needs to be emphasized is that a public bucket or a vulnerable EC2 instance or a vulnerable application installed on an EC2 instance is not a risk created by Amazon or any cloud provider but rather a misconfiguration caused by the owner who is responsible for maintaining his AWS resources. The existing vulnerabilities cannot be mitigated completely but we can try to avoid to some extent. One such efficient way is to create custom policies. In organizational or industrial level, this kind of policies are managed by third parties who mostly deny permission to create misconfigured buckets in order to ensure that proper bucket policies are applied. There are also few third party tools that can monitor your AWS account and alert you of any misconfigurations or potential attack surfaces. In cases where there are no third parties to monitor the policies, the individual user himself can perform the service compliance check and understand the level of security for his AWS resources.

V. METHODOLOGY

A. *Exploiting S3 buckets*

The following is the methodology that we are going to follow to perform penetration testing on AWS S3 bucket:

Phase 1 – Information gathering

During this phase, we have researched and gathered information about our target attack surface and developed scope of our project. We have also learned more about misconfigured S3 buckets, their vulnerabilities and how to exploit them.

Phase 2 – Modeling/Planning

During this phase, we have set up an attack environment such as configuring the kali linux machine, creating vulnerable s3 buckets, setting up an EC2 instance to capture metadata etc.

Phase 3 – Vulnerability scanning

During this phase, we have searched for the desired vulnerable bucket using a scanning tool called AWSBucketDump by providing a list of keywords as an input file.

Phase 4 – Exploitation

During this phase, we exploited the bucket by injecting a malicious JavaScript code into the bucket so that any web application that tries to access contents from our bucket will be exploited as well. We were also able to capture metadata such as access key ID and secret access key.

Phase 5 – Post-exploitation

After running the above mentioned attacks, we analyzed that strength of our S3 buckets. As we were able to successfully launch our attacks, we can say that the strength of S3 buckets is not very high. Security defenses such as encryption methods, strict access control policies etc can be enforced to safeguard S3 buckets. We have also used SOPHOS server to test the security of our S3 bucket and the compliance was found to be less than 50% with 10 critical alerts.

Phase 6 – Reporting

Though various security mechanisms are available to protect S3 buckets, it is still the user responsibility to be aware of current ongoing cyber attacks on AWS resources. It is our responsibility to ensure that our buckets are securely configured.

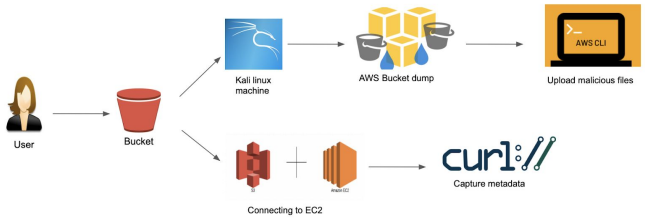

Fig 1. Methodology for exploiting S3 buckets

B. *Case study on Exploiting EC2 instances*

The following is the methodology that we are going to follow to perform penetration testing on AWS EC2 instances:

Phase 1 - Information gathering

During this phase, we have studied and gathered common vulnerabilities that exist in our target EC2 instances which helped in developing the scope of our project.

Phase 2 - Modeling/Planning

During this phase, we have set up an attack environment by installing vulnerable Windows EC2 instances with poor security configurations and also installing a vulnerable Jenkins server with default admin credentials.

Phase 3 - Vulnerability scanning

During this phase, we have searched for vulnerabilities in our Jenkins server using metasploit tool.

Phase 4 - Exploitation:

During this phase, we exploit the vulnerable modules in Jenkins server using metasploit. Out of all the vulnerable modules listed in the above step, we choose any one module to exploit. To perform this, we have to set up the target RHOST with our windows EC2 IP, RPORT as port 8080 which is commonly used to connect to our Jenkins server, username and password as admin.

Phase 5 - Post-exploitation

After we ran the attack, we were able to successfully complete the exploitation. However, the target session which is used to gain access to the script console could not be created. After studying and analyzing the Jenkins framework, it was understood that this vulnerability has been fixed. We have tried other variations of the exploit such as installing older versions of Jenkins and Windows EC2 instance. However, the target session could not be created due to fixing the vulnerability.

Phase 6 - Reporting

Though our Windows EC2 instance couldn't be a victim of our exploit, it has many other vulnerabilities which can still be exploited. As the cloud provider is only responsible for providing the services, it is the responsibility of the consumer to ensure not to install vulnerable applications on their VMs as well as keeping their VMs updated with latest software patches.

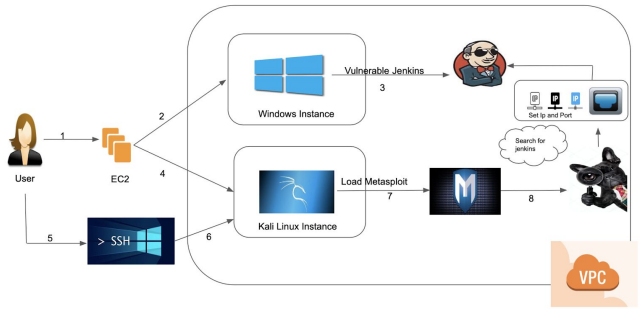

Fig 1. Methodology for exploiting EC2 instances

Conclusion

Security is considered as one of the major aspects of any industry. When choosing a platform for storing our data, we need to make sure that we follow proper methodologies and techniques to secure our data, the responsibility not only lies with the service provider but also with the individual who is hosting the data. The security threats not only come from outside, they might even occur internally or by the owner of the data.There will always be a scope for security breach that can occur due to ignorance, negligence or lack of knowledge. In the cloud environment, one of the major insecurity relies within the storage buckets where there are chances of misconfigurations which can lead to critical issues when such a bucket is compromised. This vulnerability resides in giving public access as public buckets can be accessed by anyone on the internet if proper security policies were not implemented and public data in a private bucket can also be exposed to vulnerabilities.In this project, we tried to exploit the vulnerability in the misconfigured S3 bucket. We achieved this by using some already existing tools such as AWSBucketDump which helped to detect all the currently running vulnerable buckets. We performed a case study on exploiting EC2 instances, by using a kali linux machine and creating an SSH tunnel to the EC2 instance that contains vulnerable applications and tried to exploit. It was observed that even though the application's bug has been fixed, we are still able to complete the exploitation. Since the bug has already been fixed, session to the script console could not be established. This paper also suggests few mitigations and security mechanisms that can be taken into practice in order to avoid such situations.

References

[1] Caudill, B. (2018, July 06). Penetration Testing Amazon Web Services (AWS). Retrieved December 17, 2020, from https://rhinosecuritylabs.com/penetration-testing/penetration-testing-aws-storage/

[2] 3 Big Amazon S3 Vulnerabilities You May Be. (n.d.). Retrieved December 17, 2020, from https://cloudsecurityalliance.org/blog/2020/06/18/3-big-amazon-s3-vulnerabilities-you-may-be-missing/

[3] Pandey, K. (2020, January 28). How To Secure S3 Buckets Effectively. Retrieved December 17, 2020, from https://medium.com/panther-labs/how-to-secure-s3-buckets-effectively-9c1a3a7178bb

[4] Caudill, B. (2018, July 06). AWS Security Vulnerabilities and Attack Vectors. Retrieved December 17, 2020, from https://rhinosecuritylabs.com/cloud-security/aws-security-vulnerabilities-perspective/

[5] Jordanpotti. (n.d.). Jordanpotti/AWSBucketDump. Retrieved December 17, 2020, from https://github.com/jordanpotti/AWSBucketDump

[6] How I hacked a whole EC2 Network during a Penetration Test. (2019, May 17). Retrieved December 17, 2020, from https://www.secsignal.org/en/news/how-i-hacked-a-whole-ec2-network-during-a-penetration-test/

[7] Walikar, R. (2019, December 03). Getting shell and data access in AWS by chaining vulnerabilities. Retrieved December 17, 2020, from https://blog.appsecco.com/getting-shell-and-data-access-in-aws-by-chaining-vulnerabilities-7630fa57c7ed