

null – The Open Security
Community

Common Cloud Security Issues AWS Edition



Table of Contents

About null	1
Acknowledgment	1
Abstract	2
Overview	2
List of Issues	3
1. Misconfigured S3 Buckets:	3
2. EBS Snapshots Available to the public	6
3. Subdomain Takeover on S3	7
4. Subdomain Takeover on Name Server (NS)	8
5. IAM Issues	9
6. Poorly configured Security groups	10
7. Lack of Logging	11
8. Over-use of Public Subnets	12
9. Server-Side Request Forgery (SSRF)	13
10. Insecure Lambda	14
11. Misconfigured AWS Cognito Attributes	17
12. Insecure S3 upload policy	18
13. Misconfigured Reverse Proxy	19
14. S3 Directory traversal	20
15. Exposed Origin Servers IP Address	21
16. Lack of DDos Protection	22
17. Publicly available RDS Snapshots	24



About null

null - The open security community is one of the most active and vibrant communities of cybersecurity professionals. Started with the simple idea of providing a knowledge-sharing platform to cybersecurity professionals, null has grown many folds. Currently, null has 20+ active chapters and organizes multiple security events for aspiring cybersecurity professionals. null is about spreading information security awareness. All our activities such as null Monthly Meets, null Humla, null Bachaav, null Puliya, null Job Portal are for the cause of that.

null is now focusing on contributing to enterprise security. Working on many projects to collaborate with enterprises, we conduct activities such as:

- Defining security frameworks and standards for producing security guidelines in the upcoming IT technology like Cloud, Blockchain/Cryptography, IoT, AI/ML, and many more.
- Developing tools and methodologies in order to secure the products and infrastructure based on the above-mentioned technologies.
- Initiating multiple new security projects and publish research papers.

This white paper is one small effort to contribute to achieving the above objectives.

Acknowledgment

On behalf of null - The open security community, we would like to thank the authors of this white paper, who have contributed their precious time and efforts in publishing this paper.

Authors:

- Raghav Rao : <https://www.linkedin.com/in/raghav-rao/>
- Sagar Yadwad : <https://www.linkedin.com/in/sagar-yadwad-cissp-79711123/>
- Mayank Arora : <https://www.linkedin.com/in/connect2mayank/>
- Deep Shankar Yadav : <https://www.linkedin.com/in/deepshankaryadav/>
- Vitthal Shinde : <https://www.linkedin.com/in/vitthal-shinde-7447699a/>

Project Coordinator:

- Murtuja Bharmal : <https://www.linkedin.com/in/murtujabharmal/>

Abstract

Cloud technologies empower organizations to build and run scalable applications in modern, dynamic environments such as. Cloud Services, Containers, cloud functions (serverless), service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach. Cloud-Native Applications is a fundamentally new and exciting approach to designing and building software. However, it also raises an entirely new set of security challenges. For example, when you move to a microservice model, end-to-end visibility, monitoring, and detection become more complex and challenging to execute, and security operations and management become hectic.

This white paper aims to deliver an awareness and reference document to highlight the most common AWS security issues.

Overview

The AWS cloud provides a shared responsibility model. AWS manages cloud security for its own infrastructure, while your organization is responsible for securing your own data and workloads. Amazon provides a range of security services and features, including encryption, key management and identity, and access management (IAM), to help you implement your organization's security policies.

Another important aspect of security is compliance standards and regulations, one misstep here can be costly for your organization. Amazon's infrastructure is certified for almost every compliance standard in the world. However, this doesn't mean the workloads you deploy on Amazon will be compliant as well. You must be mindful of your compliance obligations, and use the tools provided by Amazon to enforce the required security and privacy controls.

With the help of this whitepaper we have tried to cover the most common security issues that may occur in an AWS environment.

List of Issues

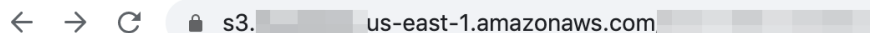
1. Misconfigured S3 Buckets:

An Amazon S3 bucket is a public cloud storage resource available in Amazon Web Services (AWS) Simple Storage Service (S3), an object storage offering. Amazon S3 buckets, which are similar to file folders, store objects, consists data and its descriptive metadata.

Attacker's View:

- S3 buckets are accessible to any authenticated user
- S3 buckets are accessible to the public
- S3 Bucket has Global DELETE Permissions enabled via bucket policy
- S3 Bucket has Global GET Permissions enabled via bucket policy
- Write/upload files to the S3 bucket
- S3 bucket reveals ACP/ACL

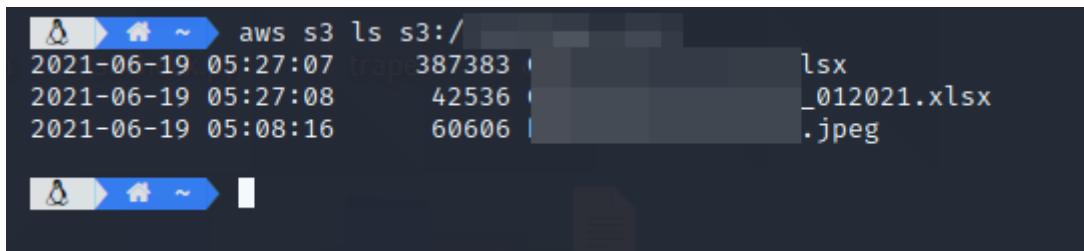
Attack Scenario:



This XML file does not appear to have any style information associated with it. The document tree is shown below.

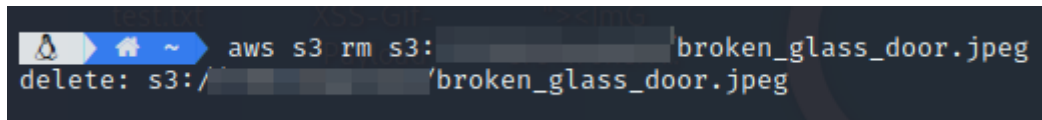
```
▼<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>[redacted]</Name>
  <Prefix/>
  <Marker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>true</IsTruncated>
  ▼<Contents>
    [redacted]
    <Size>35960</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
```

Figure 1 S3 Bucket left Open

A terminal window showing the command 'aws s3 ls s3:/' and its output. The output lists three files with their last modified dates, sizes, and names.

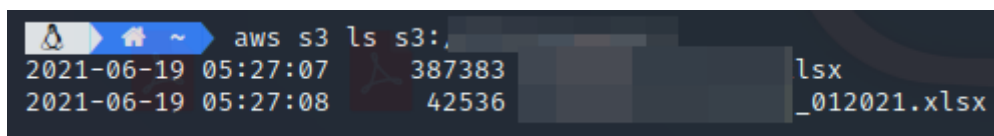
```
aws s3 ls s3:/
2021-06-19 05:27:07 387383 [redacted]lsx
2021-06-19 05:27:08 42536 [redacted]_012021.xlsx
2021-06-19 05:08:16 60606 [redacted].jpeg
```

Figure 2 Attacker Able to List Bucket Content

A terminal window showing the command 'aws s3 rm s3:[redacted]broken_glass_door.jpeg' and its output. The output confirms the deletion of the file.

```
aws s3 rm s3:[redacted]broken_glass_door.jpeg
delete: s3:[redacted]broken_glass_door.jpeg
```

Figure 3 Removing a file from S3 Bucket

A terminal window showing the command 'aws s3 ls s3:/' and its output. The output now only lists two files, confirming that the 'broken_glass_door.jpeg' file has been removed.

```
aws s3 ls s3:/
2021-06-19 05:27:07 387383 [redacted]lsx
2021-06-19 05:27:08 42536 [redacted]_012021.xlsx
```

Figure 4 Confirming that the file is deleted from the S3 bucket

All S3 Buckets URLs resembles this naming convention:

<https://name.s3.amazonaws.com>

Defender's View:

You can use Access Analyzer for S3 to review buckets with bucket ACLs, bucket policies, or access point policies that grant public access. Access Analyzer for S3 alerts you to buckets that are configured to allow access to anyone on the internet or other AWS accounts, including AWS accounts outside of your organization. For each public or shared bucket, you receive findings that report the source and level of public or shared access.

Note: By default, all Amazon S3 resources—buckets, objects, and related subresources (for example, lifecycle configuration and website configuration)—are private.

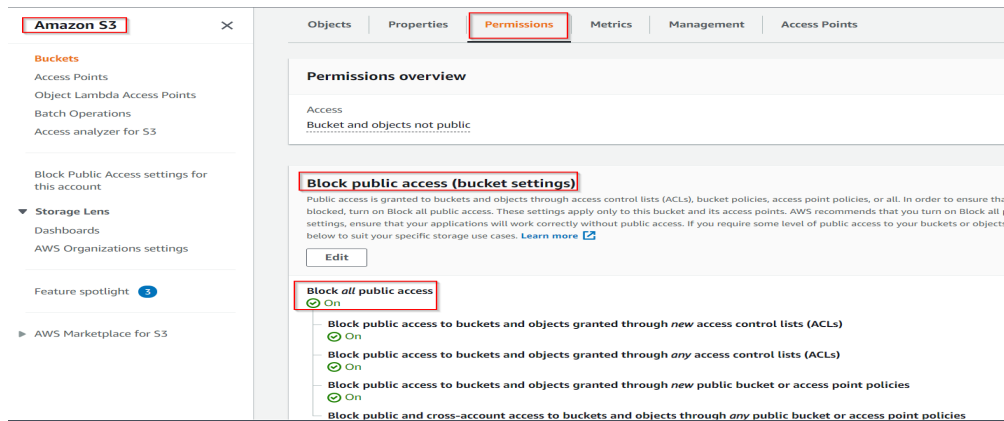


Figure 5 setting granular permission on S3

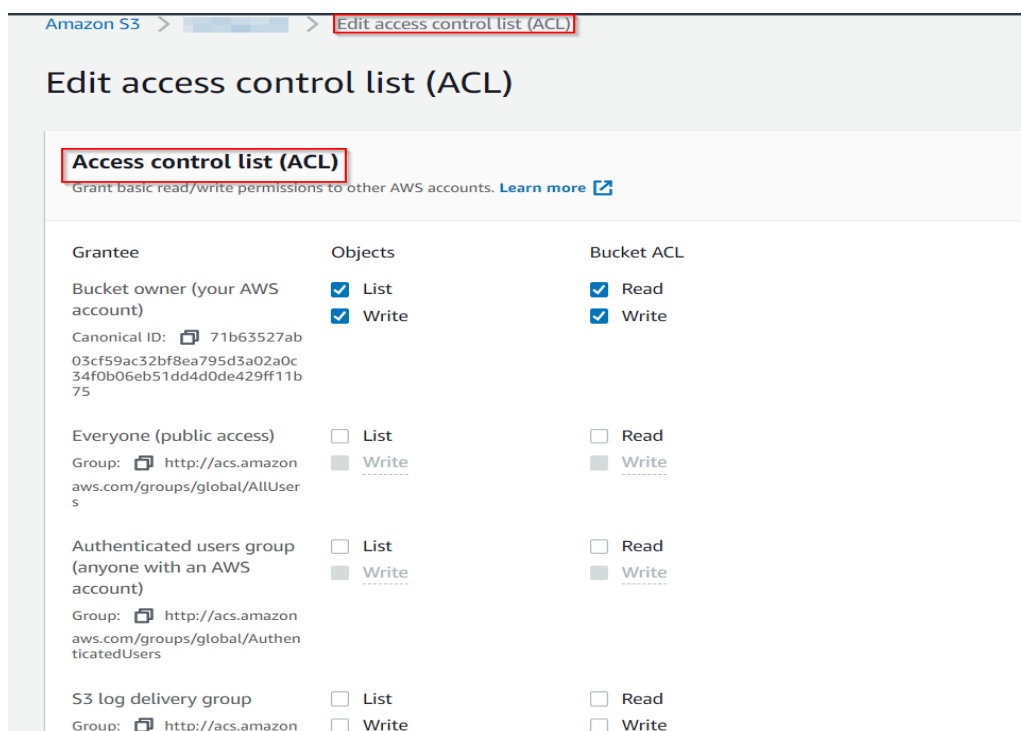


Figure 6 ACL settings on S3

Tools:

- <https://github.com/sa7mon/S3Scanner>
- <https://github.com/toniblyx/my-arsenal-of-aws-security-tools#s3-buckets-auditing>

Reference:

- <https://docs.aws.amazon.com/AmazonS3/latest/userguide/access-analyzer.html>

2. EBS Snapshots Available to the public

An EBS snapshot is a point-in-time copy of your Amazon EBS volume, which is lazily copied to Amazon Simple Storage Service (Amazon S3). EBS snapshots are incremental copies of data. This means that only unique blocks of EBS volume data that have changed since the last EBS snapshot are stored in the next EBS snapshot.

Attacker's View:

Search for unencrypted publicly exposed EBS Snapshots and mount the snapshot to your EC2 Instance and look for sensitive information, Like Credentials, Metadata URL Failures, etc.



			US East (Ohio)	
<input type="checkbox"/>	shadow_8b0ed108a45267d1f5365da665d9dfd2_vol-04d	Jan 15, 2020 11:02:57 AM GMT-0700	995.0 B	Standard
<input type="checkbox"/>	shadow_8ba70862be90ce222cbe0e0708ae0e54_vol-03	Jan 15, 2020 10:59:32 AM GMT-0700	804.0 B	Standard
<input type="checkbox"/>	shadow_8cdb73aebad0e0f4a5d1933e83171ee6_vol-00	Jan 15, 2020 11:08:49 AM GMT-0700	896.0 B	Standard
<input type="checkbox"/>	shadow_8d455024d621d92ab620a7715299be5_vol-03	Jan 15, 2020 11:05:55 AM GMT-0700	850.0 B	Standard
<input type="checkbox"/>	shadow_926828ca3bf10643fbcf95da8e06a884_vol-0d4	Jan 15, 2020 11:03:06 AM GMT-0700	850.0 B	Standard
<input type="checkbox"/>	shadow_932e0147af322f57d2459c7363989365_vol-0b4	Jan 15, 2020 10:57:04 AM GMT-0700	850.0 B	Standard
<input type="checkbox"/>	shadow_944d1b8579994529c0c2593088f6c2_vol-0b5	Jan 15, 2020 11:08:27 AM GMT-0700	850.0 B	Standard
<input type="checkbox"/>	shadow_947bf87649f97ab5f7139a997a45e5ff_vol-04c1	Jan 15, 2020 11:08:09 AM GMT-0700	850.0 B	Standard
<input type="checkbox"/>	shadow_94ac3b430e6654318bc793f87f474919_vol-0d9	Jan 15, 2020 11:07:21 AM GMT-0700	845.0 B	Standard

Figure 7 EBS snapshot publicly accessible

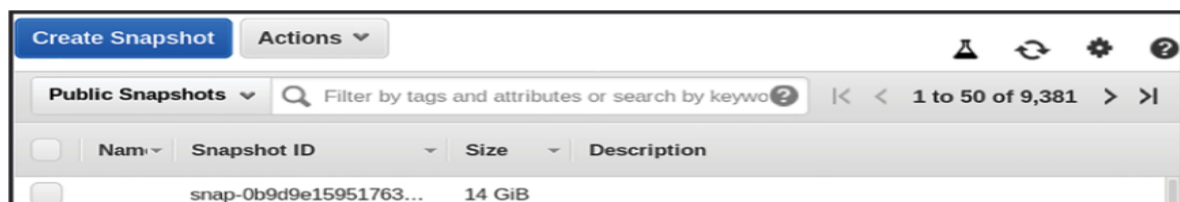


Figure 8 Compromised Public EBS Snapshot

Defender's View

- Take down the snapshot
- Rotate the credentials

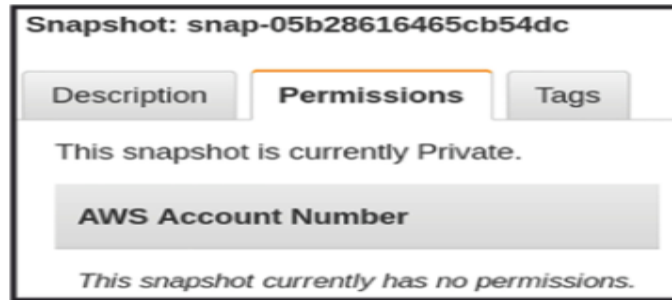


Figure 9 EBS snapshot permission set to Private from Public

Note: EBS Volumes are private by default

Tool:

- <https://github.com/bishopfox/dufflebag>

Reference:

- <https://labs.bishopfox.com/tech-blog/dufflebag-uncovering-exposed-ebs>

3. Subdomain Takeover on S3

Subdomain takeover is the process of registering a non-existing domain name to gain control over another domain.

Attacker's View:

- dig cname <Enter the sub-domain>
- Visit cname URL provided by dig command
- If you notice the message "The specified bucket does not exist" then this particular s3 is vulnerable.
- If you notice the message "Access denied", then this particular s3 is not vulnerable

Defender's View:

- Remove the dangling CNAME

Tool:

- <https://github.com/guptabless/unclaim-s3-finder>

Reference

- <https://github.com/EdOverflow/can-i-take-over-xyz>

4. Subdomain Takeover on Name Server (NS)

Subdomain takeover is the process of registering a non-existing domain name to gain control over another domain.

Attacker's View:

- Run the following command
- `host -t NS <domain name>` or `dig ns <domain name>@8.8.8.8 +nostats`
- Notice that the NS points to AWS DNS
- If DNS zone is deleted in AWS, however, NS record points to AWS then one would get a status "Serverfail"
- Create a DNS zone in AWS and NS can be taken over

Defender's View:

- Remove the dangling NS pointer

Tool:

- <https://github.com/shivsahni/NSBrute>

Reference:

- <https://0xpatrik.com/subdomain-takeover-ns/>

5. IAM Issues

AWS Identity and Access Management (IAM) enables you to manage access to AWS services and resources securely. Using IAM, you can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources.

Attacker's View:

- Get an initial foothold of the cloud with leaked credentials or brute force or social engineering
- Get the credentials by targeting web vulnerabilities like SSRF, XXE, etc in AWS.
- AWS Consoler - use it to get access to AWS console and recon through the cloud using GUI to get a better idea of the environment.
- https://github.com/NetSPI/aws_consoler
- Generate a link to console from valid credentials
- `aws_consoler -a ASIAXXXX -s SECRETXXXX -t TOKENXXXX`
- then using the credentials to enumerate the permissions available to those credentials.
- Ultimately escalating the privileges in the cloud environment, exfiltrate data, and achieve persistence depending on the IAM policy attached to the compromised user.
- escalate the privilege by targeting iam over permissive policies/ iam misconfig.

Defender's View:

- Enable two-factor authentication (2FA) for every user and IAM role.
- Add random strings to usernames and role names to make them more difficult to guess.
- Log and monitor all the identity authentication activities.
- Remove inactive users and roles to reduce the attack surface.
- Wildcards - Wildcards are easy to use and are a quick way to make things “just work,” but they should be avoided wherever possible. Think of wildcards like the “chmod 777” of the cloud world. Just because you can, doesn’t mean you should.
- Unique roles - Each application should have their own unique role, and roles should not be reused. Like the wildcards, reusing roles can be an easy way to circumvent the hassle of creating new, scoped down, permissions, but scoping access to only exactly what a resource needs will ensure that you can limit the blast radius of any compromised resource.

- Configure conditionals where possible - Using IAM conditions is a great way to ensure that permissions are only used where they're supposed to. Configuring these permissions can be tedious, but the time spent will pay dividends in your "defense in depth" strategy.

Tool:

- <https://github.com/andresriancho/enumerate-iam>
- <https://github.com/RhinoSecurityLabs/pacu>

Reference:

- <https://rhinosecuritylabs.com/aws/aws-privilege-escalation-methods-mitigation/>

6. Poorly configured Security groups

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. When you launch an instance in a VPC, you can assign up to five security groups to the instance. Security groups act at the instance level, not the subnet level. Therefore, each instance in a subnet in your VPC can be assigned to a different set of security groups.

Attacker's View:

- perform recon on the cloud for services publicly accessible endpoints
- enumerate the ec2 instances if publicly accessible for what ports are open.
- scan for vulnerable service running
- check for any misconfiguration and public exploit available.
- exploit the vulnerable service and get an initial foothold by getting IAM credentials.

Defender's View:

- Ensure EC2 SGs Do Not Have Large Range of Ports Open
- Never Keep Unattached Security Groups and Limit Modifications to Only Certain Roles
- Use ELB's SGs Wisely to Restrict EC2s' Access to the Internet
- Do Not Ignore Outbound Rules of SG; Set Restrictions Decisively

Tool:

- https://github.com/arkadiyt/aws_public_ips

7. Lack of Logging

Log files are invaluable in the event of Security events and forensic investigation. AWS CloudTrail is a service that enables governance, compliance, operational auditing, and risk auditing of your AWS account. With CloudTrail, you can log, continuously monitor, and retain account activity related to actions across your AWS infrastructure. CloudTrail provides the event history of your AWS account activity, including actions taken through the AWS Management Console, AWS SDKs, command-line tools, and other AWS services. This event history simplifies security analysis, resource change tracking, and troubleshooting.

Attacker's view:

- TrailBlazer can be used to model attacks in your environment to aid in testing monitoring. Due to the way that TrailBlazer makes calls, you can safely use TrailBlazer in your production environment to model an attacker in your environment.

Defender's view:

- If logging is not enabled, then most of the security events are missed and there will be very little visibility
- Implement centralized logging as a best practice.

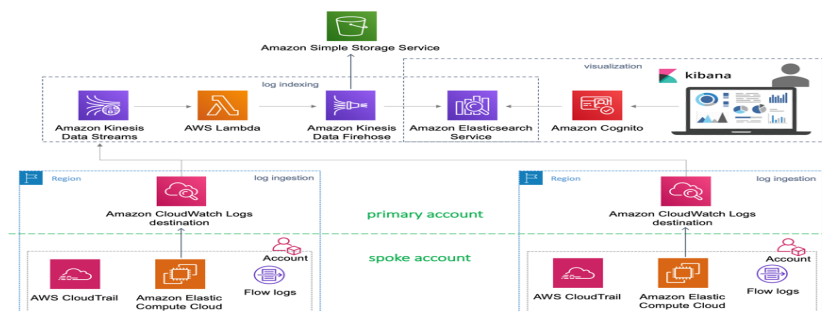


Figure 10 AWS centralized logging

Tool:

- <https://github.com/willbengtson/trailblazer-aws>

Reference:

- <https://aws.amazon.com/solutions/implementations/centralized-logging/>
- <https://docs.aws.amazon.com/whitepapers/latest/aws-security-incident-response-guide/logging-and-events.html>

8. Over-use of Public Subnets

Using default VPC subnets are dangerous as these are routed to Internet Gateways making them accessible via the internet.

Attacker's view:

Reaching critical / Internal applications and databases via the internet

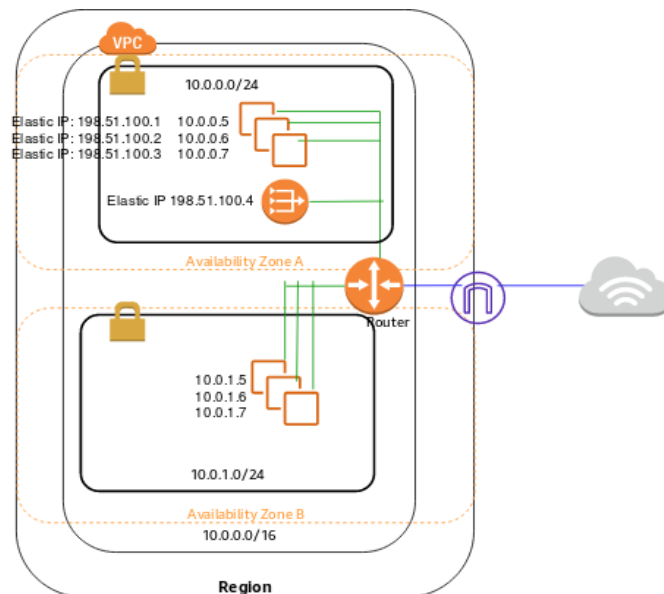


Figure 11 Public-Private Subnet Topology

Defender's view:

- Public subnets are suitable for blogs or simple websites, but not critical applications or databases. Instead, use private subnets and ensure all subnets are properly configured for the security needs of the assets involved.

Tool:

- https://github.com/arkadiyt/aws_public_ips

Reference:

- https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Scenario2.html
- <https://cobalt.io/blog/aws-cloud-security>

9. Server-Side Request Forgery (SSRF)

SSRF (Server Side Request Forgery) is the attack that allows an attacker to send a request on behalf of the server, it is a web to network level attack that compromises the internal machines, basically, a request being getting forged and sending it to the victim's server. The risk of SSRF may depend on how much information is being accessed, from low to critical

Attacker's view:

- Finding web application hosted on AWS ec2 .
- access web application to find SSRF vulnerability .
- exploit the SSRF vulnerability and access the metadata service to get IAM credentials.

Services Vulnerable to SSRF Attacks

- Elasticsearch SSL/TLS is not enabled by default, and data can be read in plaintext from HTTP REST APIs.
- Consul allows reading and writing to its data via HTTP API.
- MongoDB comes with an HTTP REST API by default.
- EC2 Instance Metadata Service (IMDS) is the vector that ultimately led to the Capital One breach. EC2 Instances have access to a universal.

Defender's view:

- Enable IMDSv2
- Limit access to IMDS
- Ensure proper input sanitization at application level
- Lock Down IMDS to specific users

Reference:

- <https://blog.appsecco.com/an-ssrf-privileged-aws-keys-and-the-capital-one-breach-4c3c2cded3af>
- <https://aws.amazon.com/blogs/security/defense-in-depth-open-firewalls-reverse-proxies-ssrf-vulnerabilities-ec2-instance-metadata-service/>
- <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html>
- <https://docs.aws.amazon.com/cli/latest/reference/ec2/modify-instance-metadata-options.html>

10. Insecure Lambda

Lambda function is an AWS serverless computing service designed to execute code only when needed or triggered. Once the execution is over, the computing instance that runs the code decommissions itself. You can create as many functions as the application needs for handling different tasks. Similar to vulnerabilities detected on traditional web applications, cloud applications running on serverless services can also be prone to the same security vulnerabilities.

Attacker's view:

- Inspect All visible URLs
- View the source page and look for hidden comments, API endpoints, storage locations
- Check Enabled HTTP Methods
- Manually fuzz the application and observe the responses carefully
- test for serverless security issues in the endpoint
- if found vulnerable test for if environment variables are accessible.
- check if the Lambda can reveal juicy information about the AWS account it runs on.

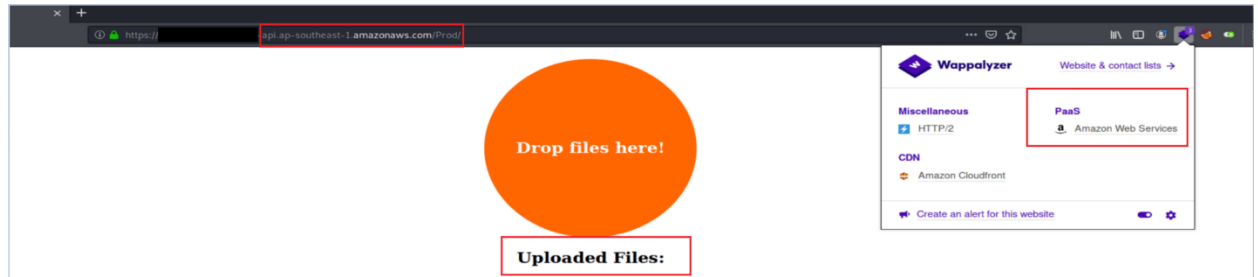


Figure 12 Fingerprinting aws service

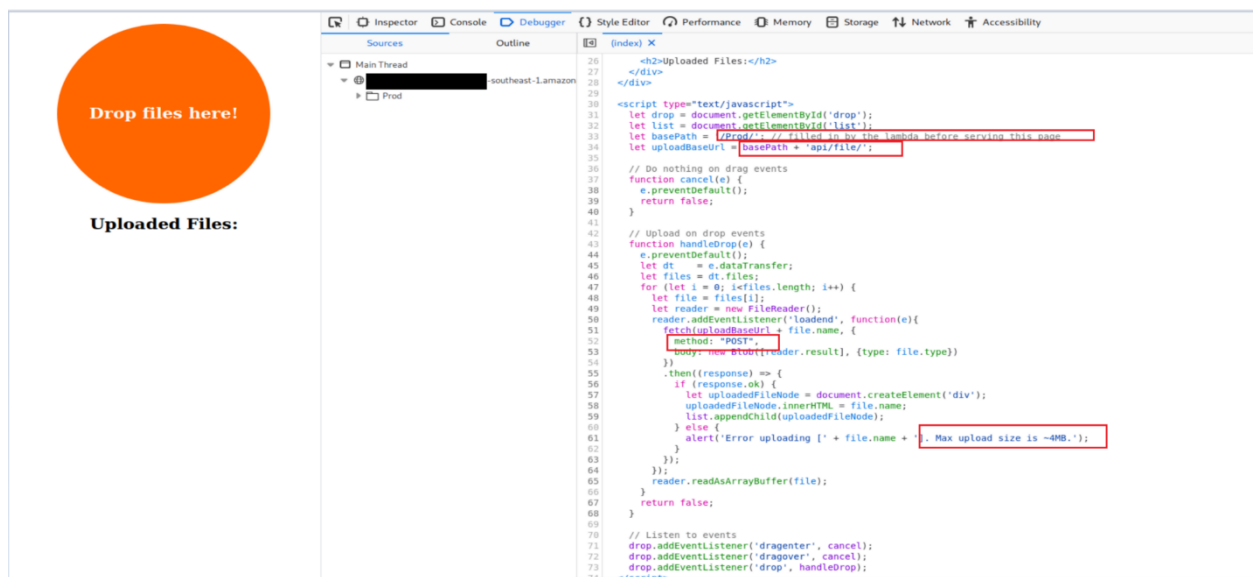


Figure 13 File Size restriction

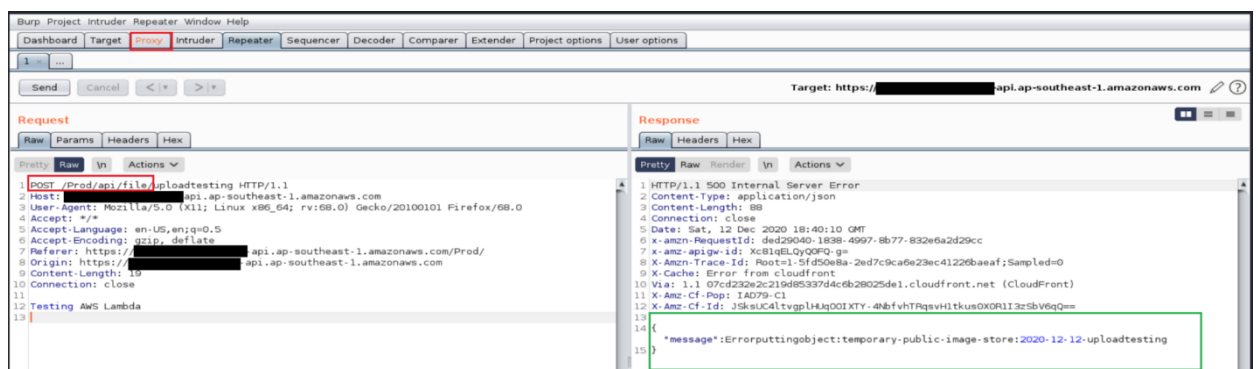


Figure 14 Malicious file upload

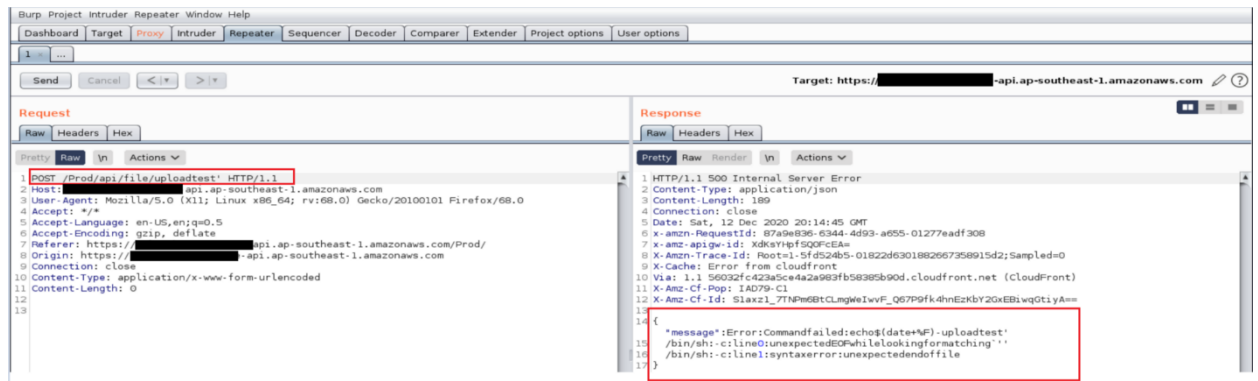


Figure 15 Error message based on our malicious input



Figure 16 Listing backend files

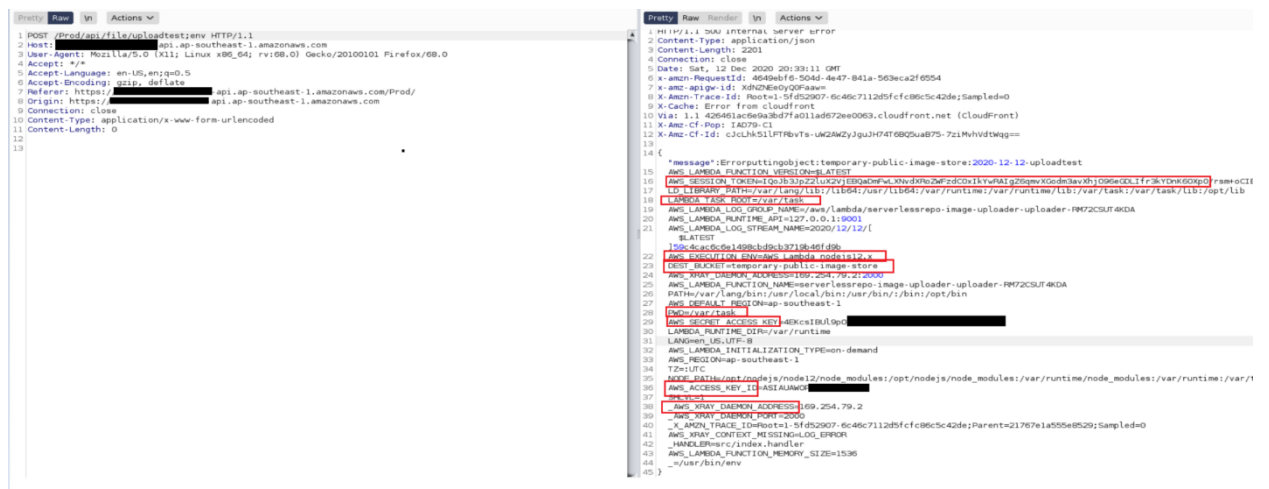


Figure 17 Listing sensitive details from environment variable

The most common types of injection flaws in serverless architectures are presented below (in no particular order):

- Operating System (OS) command injection
- Function runtime code injection (e.g. Node.js/JavaScript, Python, Java, C#, Golang)
- SQL injection
- NoSQL injection
- Pub/Sub Message Data Tampering (e.g. MQTT data Injection)
- Object deserialization attacks
- XML External Entity (XXE)
- Server-Side Request Forgery (SSRF)

Defender's view

- Use environment variables for configuration details and encrypt with KMS
- Retrieve sensitive data from DynamoDB, Secrets Manager, or similar service
- Enable encryption helpers to protect data in transit
- Configure API Gateway custom domain with SSL cert for web access
- List minimum specific actions in role policies – not “*”
- Limit outbound access to the Internet to prevent unauthorized data exfiltration
- Do not use /tmp for sensitive information
- Application security testing and source code analysis
- Ensure no vulnerable or unmaintained dependencies
- Monitor Lambda logs and investigate unusual behaviour

Reference:

- <https://github.com/puresec/sas-top-10>
- <https://www.celidor.co.uk/blog/10-steps-to-lambda-security>

11. Misconfigured AWS Cognito Attributes

Amazon Cognito is a managed authentication and authorization service by Amazon AWS that enables developers to integrate a flexible and scalable user management system into their web applications.

Amazon Cognito consists of two main things and those are:

- **User Pools:** User pools allow sign-in and sign-up functionality
- **Identity Pools:** Identity pools allow authenticated and unauthenticated users to access AWS resources using temporary credentials

Attacker's View

When an adversary observes that the AWS Cognito is being used, he/she will check for misconfigurations like self-registration features, updating custom attributes, listing and reading the contents of S3 buckets etc

The screenshot shows the 'Attributes' section in the AWS Cognito console. It is divided into two columns: 'Readable Attributes' and 'Writeable Attributes'. Both columns have a 'Scopes' section with checkboxes for 'Address', 'Email', 'Phone Number', and 'Profile'. Below these are lists of attributes with checkboxes. In the 'Writeable Attributes' column, the 'custom:role' attribute is highlighted with a red box.

Readable Attributes		Writeable Attributes	
Scopes	Address, Email, Phone Number, Profile	Scopes	Address, Profile
Attributes	address, birthdate, email, email verified, family name, gender, given name, locale, middle name, name, nickname, phone number, phone number verified, picture, preferred username, profile, zoneinfo, updated at, website	Attributes	address, birthdate, email, family name, gender, given name, locale, middle name, name*, nickname, phone number, picture, preferred username, profile, zoneinfo, updated at, website, custom:role

Figure 18 Cognito Attributes

The screenshot shows a dialog box titled 'Do you want to allow users to sign themselves up?'. It contains a link 'Learn more' and two radio button options. The option 'Allow users to sign themselves up' is selected and highlighted with a red box.

Do you want to allow users to sign themselves up?

You can choose to only allow administrators to create users or allow users to sign themselves up. [Learn more](#)

☐ Only allow administrators to create users

☒ Allow users to sign themselves up

Figure 19 Cognito sign up

Defender's View

- Remove sensitive details from server responses, including Cognito Identity Pool Id. The server must return minimal data whenever a server request is made.
- Review IAM policy attached to the unauthenticated role while configuring Amazon Cognito to ensure least privilege access.

12. Insecure S3 upload policy

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure

access to your data to meet your specific business, organizational, and compliance requirements.

Attacker's View

When testing an application if you see S3 buckets are used to store your information like Images or media files.

Try uploading the files to different directories to see if you can overwrite information.

Defender's view

Developers must always define a base path when creating a POST upload policy

13. Misconfigured Reverse Proxy

What is a Reverse Proxy?

A reverse proxy server is a type of proxy server that typically sits behind the firewall in a private network and directs client requests to the appropriate backend server. A reverse proxy provides an additional level of abstraction and control to ensure the smooth flow of network traffic between clients and servers.

Common use of reverse proxy

- Load Balancing
- Web acceleration
- Security and anonymity

Attacker's View

If the proxy is configured as a forwarding proxy, an attacker can try following things

- Open redirects
- SSRF

- CRLF Attacks
- Phishing attacks
- Stealing credentials from AWS meta-data

PS: Often times Dev-Ops team forget to limit the exposure of staging servers to search engines, which will lead to proxy servers getting exploited.

Defender's view

- Ensure proper access controls and permissions are in place

14. S3 Directory traversal

What is a S3?

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.

Pre-signed URLs: These pre-signed URLs give temporary access to the specific S3 object

Attacker's View

- When testing the application, If the application generates a pre-signed URL, then try for directory traversal attacks against those parameters

Defender's view

- The most effective way to prevent Directory traversal is to avoid passing user-supplied input to file system and API altogether
- Perform strict input validation against all the parameters

15. Exposed Origin Servers IP Address

Cloudflare is one of the fastest-growing CDN providers, that offers free and premium services to accelerate, optimize & secure websites.

Protecting the real IP address of a website is essential to prevent DDoS attacks

Attacker's View

Bypassing CDN such as cloudflare and directly accessing server IP may bypass all the protection enabled by cloudflare.

Attacker can find out source IP of website or database origin servers using publically available tools such as

Censys (<https://censys.io/domain>),

Shodan - <https://www.shodan.io/>,

IVRE - <https://ivre.rocks/>,

Zoom EYE - <https://www.zoomeye.org/>,

Security Trails - <https://securitytrails.com/> ,

Crimeflare - <http://www.crimeflare.org:82/cfs.html>

Virtual Hosts - <https://pentest-tools.com/information-gathering/find-virtual-hosts>

Defender's view

- Keeping all their subdomains on the same CDN
- Not to initiate an outbound connection based on user action
- Changing their Origin IP while configuring CDN, DNS records are many places where historical records are archived. These historical DNS records will contain the website Origin IP Using CDN.
- Restricting Direct Access to the website using IP Address
- Whitelisting IP Addresses

Tools

- <https://github.com/HatBashBR/HatCloud>
- <https://github.com/m0rtem/CloudFail>

16. Lack of DDos Protection

A Denial of Service (DoS) attack is a malicious attempt to affect the availability of a targeted system, such as a website or application, to legitimate end users. Typically, attackers generate large volumes of packets or requests ultimately overwhelming the target system. In case of a Distributed Denial of Service (DDoS) attack, and the attacker uses multiple compromised or controlled sources to generate the attack.

Defender's view

1. Reduce Attack Surface Area

One of the first techniques to mitigate DDoS attacks is to minimize the surface area that can be attacked, thereby limiting the options for attackers and allowing you to build protections in a single place. We want to ensure that we do not expose our application or resources to ports, protocols or applications from where they do not expect any communication. Thus, minimizing the possible points of attack and allowing concentrating on mitigation efforts. In some cases, you can do this by placing your computation resources behind Content Distribution Networks (CDNs) or Load Balancers and restricting direct Internet traffic to certain parts of your infrastructure like your database servers. In other cases, you can use firewalls or Access Control Lists (ACLs) to control what traffic reaches your applications.

2. Plan for Scale

The two key considerations for mitigating large-scale volumetric DDoS attacks are bandwidth (or transit) capacity and server capacity to absorb and mitigate attacks.

Transit capacity. When architecting your applications, make sure your hosting provider provides ample redundant Internet connectivity that allows you to handle large volumes of traffic. Since the ultimate objective of DDoS attacks is to affect the availability of your resources/applications, you should locate them, not only close to your end users but also to large Internet exchanges which will give your users easy access to your application even during high volumes of traffic. Additionally, web applications can go a step further by employing Content Distribution Networks (CDNs) and smart DNS resolution services which provide an additional layer of network infrastructure for serving content and resolving DNS queries from locations that are often closer to your end users.

Server capacity. Most DDoS attacks are volumetric attacks that use up a lot of resources; it is, therefore, important that you can quickly scale up or down on your computation resources. You can either do this by running on larger computation resources or those with features like more extensive network interfaces or enhanced networking that support larger volumes. Additionally, it is common to use load balancers to continually monitor and shift loads between resources to prevent overloading any one resource.

3. Know what normal and abnormal traffic is

Whenever we detect elevated traffic levels hitting a host, the very baseline is to be able only to accept as much traffic as our host can handle without affecting availability. This concept is called rate limiting. More advanced protection techniques can go one step further and intelligently only accept legitimate traffic by analyzing the individual packets themselves. To do this, you need to understand the characteristics of good traffic that the target usually receives and be able to compare each packet against this baseline.

4. Deploy Firewalls for Sophisticated Application attacks

A good practice is to use a Web Application Firewall (WAF) against attacks, such as SQL injection or cross-site request forgery, that attempt to exploit a vulnerability in your application itself. Additionally, due to the unique nature of these attacks, you should be able to easily create customized mitigations against illegitimate requests which could have characteristics like disguising as good traffic or coming from bad IPs, unexpected geographies, etc. At times it might also be helpful in mitigating attacks as they happen to get experienced support to study traffic patterns and create customized protections.

Reference :

<https://aws.amazon.com/blogs/security/how-to-protect-your-web-application-against-ddos-attacks-by-using-amazon-route-53-and-a-content-delivery-network/>

17. Publicly available RDS Snapshots

Amazon Relational Database Service is a distributed relational database service by Amazon Web Services. It is a web service running "in the cloud" designed to simplify the setup, operation, and scaling of a relational database for use in applications.

If your RDS snapshot is public, then the data which is backed up in that snapshot is accessible to all other AWS accounts. Other AWS users can not only access and copy your data but can also create a volume out of it.

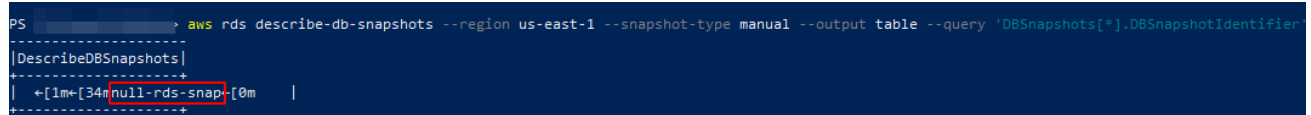
Attacker's View

Attacker can execute the following command to check the snapshots available.

```
aws rds describe-db-snapshots --region us-east-1 --snapshot-type manual --output table --query 'DBSnapshots[*].DBSnapshotIdentifier'
```

If the below command returns output "AttributeValues" set to all then instance is publicly available.

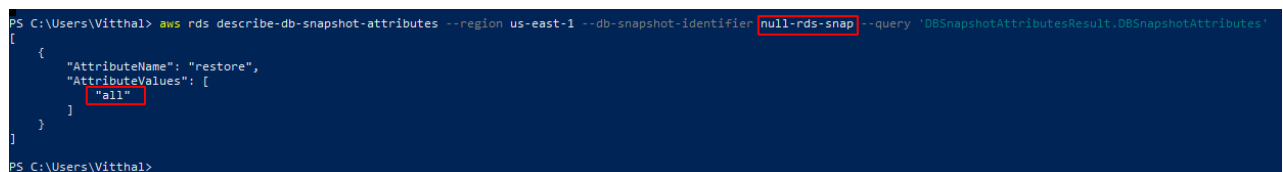
```
aws rds describe-db-snapshot-attributes --region us-east-1 --db-snapshot-identifier <name> --query 'DBSnapshotAttributesResult.DBSnapshotAttributes'
```



```
PS C:\Users\Vitthal> aws rds describe-db-snapshots --region us-east-1 --snapshot-type manual --output table --query 'DBSnapshots[*].DBSnapshotIdentifier'
DescribeDBSnapshots
+-----+-----+-----+-----+
| +[1m+[34mnull-rds-snap-[0m      |
+-----+-----+-----+-----+

```

Figure 20 RDS snapshot Identifier



```
PS C:\Users\Vitthal> aws rds describe-db-snapshot-attributes --region us-east-1 --db-snapshot-identifier null-rds-snap --query 'DBSnapshotAttributesResult.DBSnapshotAttributes'
[
  {
    "AttributeName": "restore",
    "AttributeValues": [
      "all"
    ]
  }
]
PS C:\Users\Vitthal>
```

Figure 21 RDS snapshot attributes

Defender's view

Restrict completely the public access to your RDS database snapshots and make them private (i.e. only accessible from the current AWS account)