Hacking misconfigured AWS S3 buckets: A complete guide

BY BLACKBIRD-EU · SEPTEMBER 5, 2024 · LAST UPDATED ON MARCH 6, 2025

AWS S3 (Simple Storage Service) buckets are a popular storage service used by software companies and organizations to store public as well as sensitive data. However, the implementation of this service is not always correctly done. A single missing access policy can often introduce security risks, data leaks, or other unintended consequences.

In this article, we will cover some of the most common security misconfigurations in AWS S3 buckets.

Finding & identifying AWS S3 buckets

There are several ways to find and identify AWS S3 bucket names. Below are 3 of the most effective ways of enumerating AWS S3 buckets.

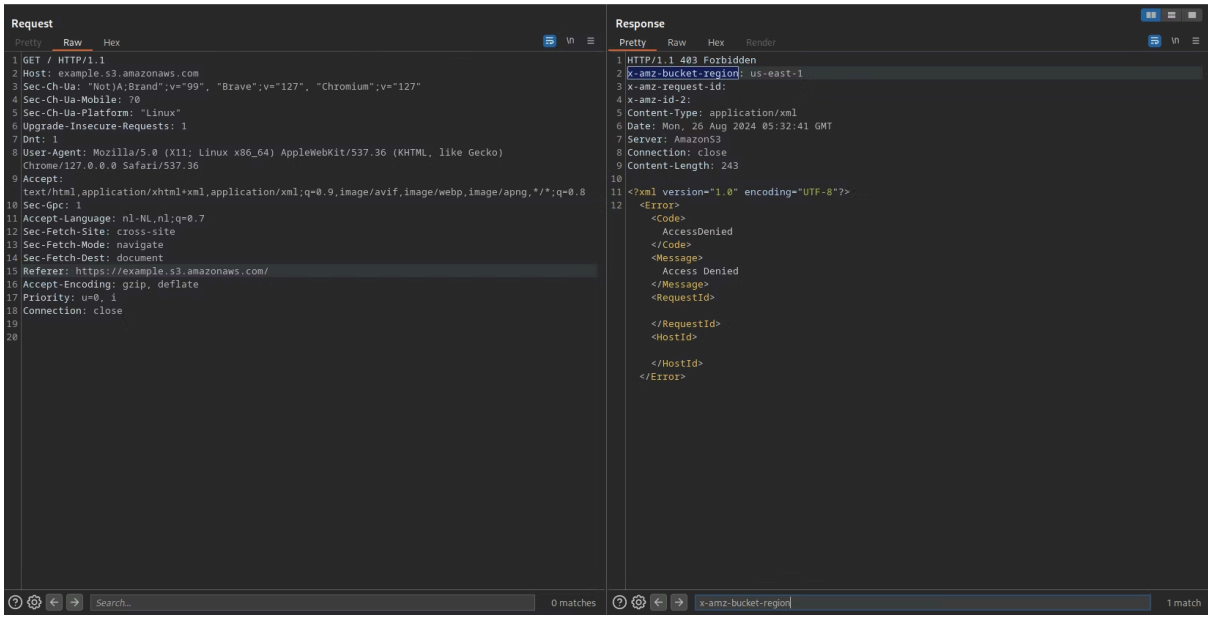
Examining HTTP responses:

One method is simply examining HTTP responses in your proxy intercepting tool. Often AWS S3 bucket references & links are included in the HTTP response to load images or other files. Search for:

```
\.s3\.amazonaws\.com\/?
```

Or search for one of the following HTTP response headers:

```
x-amz-bucket-region
x-amz-request-id
x-amz-id-2
```

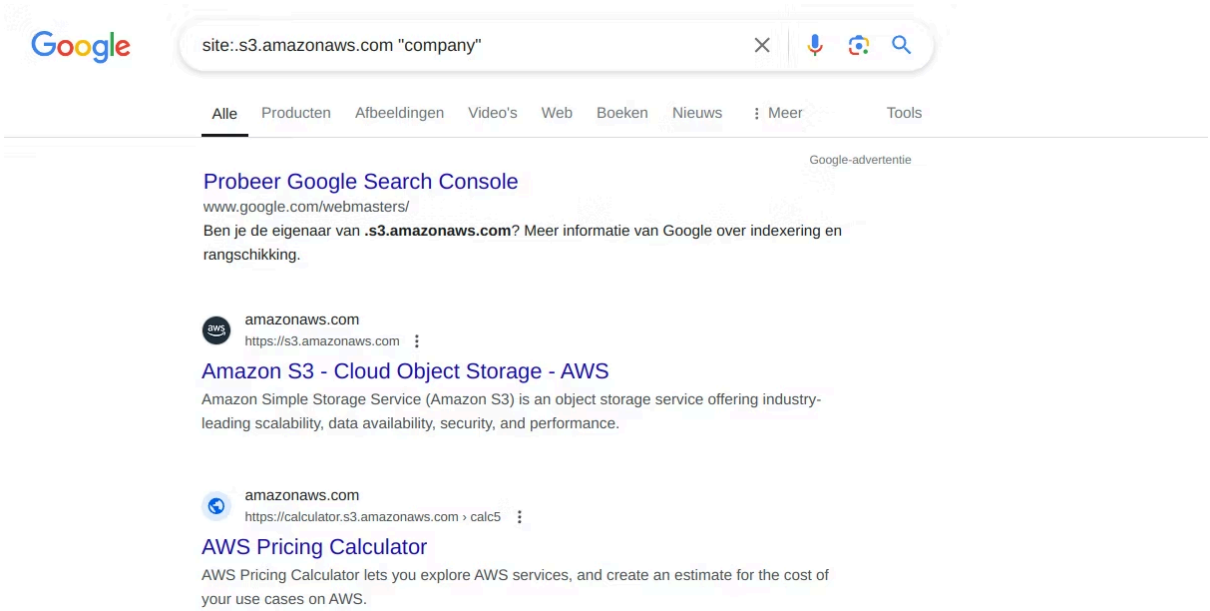


Example of an AWS S3 bucket response

Dorking:

Several popular search engines like Google, Bing, DuckDuckGo and Brave Search support search syntaxis. You can take advantage of this by specifically looking for your company and browse through indexed results.

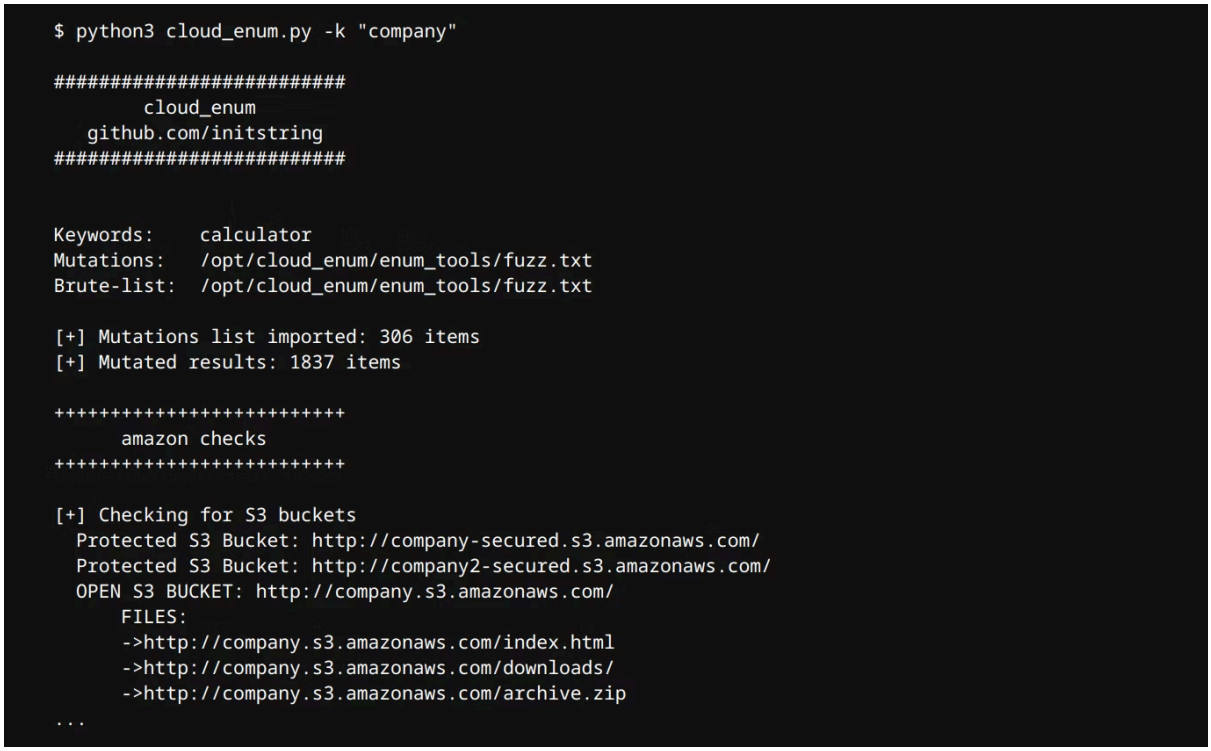
```
site:.s3.amazonaws.com "company"
```



Finding AWS S3 buckets with search engines

Bruteforcing:

You can also bruteforce common keywords your target may use as a bucket name. Automated tools such as **S3enum** and **cloud_enum** can help you enumerate AWS S3 buckets.



cloud_enum preview

Now that know how to enumerate AWS S3 buckets, we can start testing each one of them individually. At the end of this article, we will also go over a few popular open-source tools to help you automate these checks.

TIP! To test AWS S3 buckets, we will need an active AWS account and the official AWS CLI installed. Make sure to also configure your AWS CLI after successfully installing it.

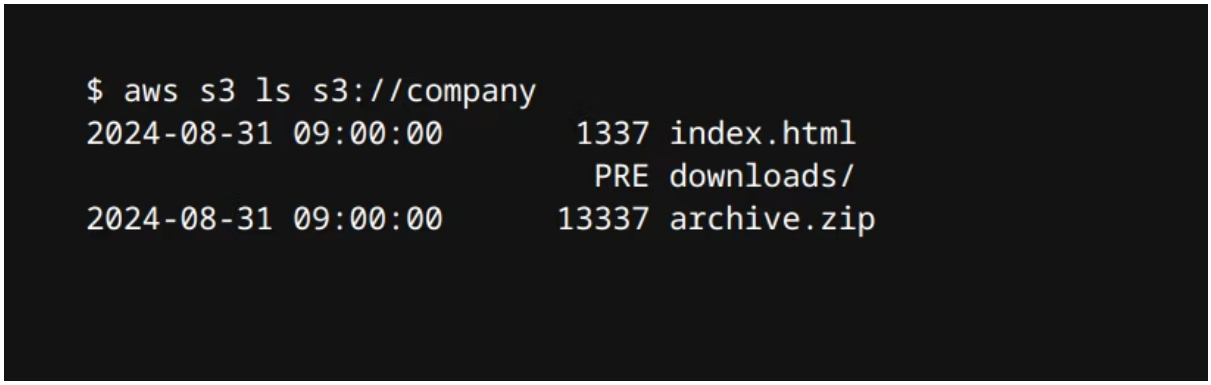
1) Testing for misconfigured list permissions in AWS S3

The most basic test we can perform on an AWS S3 bucket is testing the list permissions. Previously, AWS S3 buckets had list permissions enabled by default. But recently that changed and any newly created buckets have list permissions disabled by default. Developers and admins will now need to explicitly declare a policy to allow S3 listing.

We can however still test for list permissions on our target AWS S3 bucket by running the following command:

```
aws s3 ls s3://{BUCKET_NAME} --no-sign-request
```

You should receive the following output if list permissions are enabled:



Example output

TIP! Before reporting a potential security misconfiguration, always verify the impact of the vulnerability! Some AWS S3 buckets are meant to be public!

2) Testing for misconfigured read permissions in AWS S3

As you can see from the previous results, we could list all the contents inside a storage bucket with list permissions enabled. Access controls can also be set on individual objects within a storage bucket.

An example might be that the **index.html** file is accessible but the **archive.zip** is not. But before jumping to any conclusions, we can simply perform the test ourselves:

```
aws s3api get-object --bucket {BUCKET_NAME} --key archive.zip ./OUTPUT --no-sign-request
```

Make sure to replace the filename and bucket name in the command above.

3) Testing for misconfigured download permissions in AWS S3

To quickly check if we can download and read files, we can use the **cp** subcommand:

```
aws s3 cp s3://{BUCKET_NAME}/intigriti.txt ./ --no-sign-request
```

Make sure to replace the filename and bucket name in the command above.

4) Testing for misconfigured write permissions in AWS S3

To test for missing write permissions, we can use the following command:

```
aws s3 cp intigriti.txt s3://{BUCKET_NAME}/intigriti-ac5765a7-1337-4543-ab45-1d3c8b468ad3.txt --no-sign-request
```

Make sure to use a filename with a non-trivial name to prevent any disruption and replace the bucket name in the command above.

This operation can write and even overwrite a file, essentially deleting the old one. If bucket versioning is not enabled, the changes are permanent and permanent data loss is possible which can be catastrophic for a company.

5) Testing for read permissions on Access Control Lists (ACLs)

An Access Control List (ACL, commonly also referred to as ACP) is a predefined scheme to help manage access controls on a specific S3 bucket or an object within an S3 bucket.

These ACLs can also have misconfigured access controls and allow malicious actors to request them, essentially allowing them to take a shortcut and easily enumerate any storage buckets or objects with misconfigured access controls.

To attempt to read an ACL on your S3 storage bucket using the AWS CLI, run the following command:

```
aws s3api get-bucket-acl --bucket {BUCKET_NAME} --no-sign-request
```

Make sure to replace the bucket name in the command above.

To read an ACL on a specific object within a bucket using the AWS CLI, run the following command:

```
aws s3api get-object-acl --bucket {BUCKET_NAME} --key index.html --no-sign-request
```

Make sure to replace the filename and bucket name in the command above.

6) Testing for write permissions on Access Control Lists (ACLs)

AWS also provides you the option to overwrite any Access Control List (ACL) for an S3 storage bucket or object.

If permissions are misconfigured, it would allow a malicious actor to overwrite any current policies. This could allow a malicious actor to grant himself/herself permission to view the contents of the S3 storage bucket with no extra effort.

We can test for write permissions on ACLs by running the following command:

```
aws s3api put-bucket-acl --bucket {BUCKET_NAME} --grant-full-control emailaddress={EMAIL} --no-sign-request
```

Make sure to replace the bucket name and the email in the command above.

TIP! You do not have to always necessarily change an ACL to test for write permissions if read permissions are enabled. You can simply check the "Grants" property in the response to a read

operation and verify if any unauthorized users are allowed to perform the write operation!

7) Testing for missing file type restrictions

AWS S3 buckets are often also used for storing public data such as profile images. And there are several ways developers are used to upload data to an AWS S3:

1. Make the client upload the data to the company's API server, perform validations on the uploaded file, and finally store it on AWS S3 and return the public URL. The API server acts here as a proxy between the client and the AWS S3 bucket API.

2. Or make the client upload the data directly to AWS S3 by sending a form upload request to the AWS S3 bucket API. The API will return a link from which you can access the resource. To perform validation, you'd have to declare additional policies to the `s3:PutObject` for your AWS S3 bucket (this step is often neglected as developers aren't aware of it).

The reason why most developers select the latter approach over the former is to decrease the load on their server. However, as mentioned before, developers often forget to declare additional file type restriction policies to further control what can be uploaded.

In other words, as an attacker, you can often easily upload malicious files like SVG files to achieve stored XSS for example.

In this video by @gregxsunday, you could view another example of how developers sometimes make mistakes when integrating AWS S3:

8) Testing for S3 versioning

S3 versioning is an additional security measure to allow bucket owners to store multiple versions of the same object. If an object accidentally gets modified, or even worse deleted, admins can simply revert to the latest version and undo the unwanted action.

For that reason, AWS always recommends you enable S3 versioning. However, storing multiple versions of the same file comes with additional costs that most development teams want to prevent. Turning that option off seems reasonable but this comes with a few security issues. If a malicious user ever gets the chance to delete or overwrite files, the changes will be permanent and there will be no way of restoring the original data objects (unless a separate backup is made).

You can test if unauthorized users can check if S3 versioning is enabled by running the following command:

```
aws s3api get-bucket-versioning --bucket {BUCKET_NAME} --no-sign-request
```

Automated tools

Enumerating and testing all your targets' S3 buckets individually can be a tedious task, especially when your target makes use of several S3 buckets. Fortunately for us, there are several open-source tools that we can make use of.

Here are a few open-source tools listed that can help with identifying and exploiting AWS S3 buckets.

S3enum

S3enum is a fast and stealthy AWS S3 bucket enumeration tool written in Golang and used by bug bounty hunters and penetration testers to enumerate AWS S3 buckets.

https://github.com/koenrh/s3enum

cloud_enum

cloud_enum is an extensive OSINT tool to help bug bounty hunters and penetration testers enumerate cloud buckets such as AWS S3, GCP buckets and even Azure storage buckets.

https://github.com/initstring/cloud_enum

LazyS3

LazyS3 is a Ruby script that is capable of enumerating and identifying potential S3 buckets that belong to your target.

https://github.com/nahamsec/lazys3

AWS Extender

AWS Extender is a Burpsuite plugin (Professional edition only) to help you test for permissions on AWS S3, Google Cloud Provider storage buckets and Azure Storage Containers.

https://github.com/VirtueSecurity/aws-extender

Nuclei

Nuclei is a powerful template-based scanner that is capable of identifying and testing several permissions and access control lists (ACLs) on AWS S3 buckets using custom templates.

https://github.com/projectdiscovery/nuclei

Conclusion

You've probably already come across an AWS S3 bucket, maybe even a misconfigured one too, and if you ignored them before, we hope this article shines some light on the most common security misconfigurations present in this storage bucket service.

So, you've just learned something new about AWS S3 security misconfigurations... Right now, it's time to put your skills to the test! You can start by practicing on vulnerable labs or... browse through our 70+ public bug bounty programs on Intigriti and who knows, maybe earn a bounty on your next submission!

START HACKING ON INTIGRITI TODAY