Amazon Cloud Security Testing 101

Federico De Meo, Security Consultant, MindedSecurity

OWASP Italy Day
Milano, 14th March 2019
• Once upon a time ...
• What are clouds made of?
• Amazon AWS
• S3
• EC2
• CloudFront
• Cognito
Why Amazon??

We encounter it a lot ... everywhere!

(not sponsored in any way, shape or form)

One upon a time ...
... while today!

Your stuff is ... somewhere in there!

amazon web services

Google Cloud Platform

Microsoft Azure
What are clouds made of?

A place where you can **do** stuff

A place where you can **store** stuff
Amazon AWS

- Simple Storage Service (S3)
- Elastic Compute Cloud (EC2)
- CloudFront
- Cognito
Simple Storage Service (S3)

Is a container of stuff ...possibly sensitive stuff!!
Simple Storage Service (S3)

Defense contractor posts data in Amazon cloud
[Updated]

Booz Allen Hamilton engineer posted on the internet thousands of gigabytes of customer data that it had stored using Amazon’s Simple Storage Service (S3), according to the reports.

Verizon partner data breach exposes millions of customer records

Accessed through an unprotected Amazon S3 storage server.

Skyhigh Discovers GhostWriter: MITM Exposure In Cloud Storage Services
S3 Security Testing

Acquire target...

...FIRE!!!
S3 Buckets Identification

**HTML inspection**: find buckets directly in HTML

![HTML Code Example](https://example.com/html-code.png)

**Brute-force\educated guessing**: your company name is aegea?

Your bucket might be called **aegea[-something]**

**Google Dork**: let Google do it for you.

Technique that uses Google Search and other Google applications to find security

- Inurl: s3.amazonaws.com/legacy/
- Inurl: s3.amazonaws.com/uploads/
- Inurl: s3.amazonaws.com/backup/
- Inurl: s3.amazonaws.com/mp3/
S3 Buckets Identification

DNS Caching: let others do it for you

Services maintaining a cache of Domain ↔ IP address

You can perform any type of search and hopefully get the bucket you’re looking for.
**S3 Buckets Identification**

**Bing reverse IP:** who said Microsoft is useless?  
... let Bing do it for you

Search for an S3 ip  
and you’ll get the bucket associated if they appear somewhere
S3 Permissions Testing

READ

aws s3 ls s3://[bucketname] --no-sign-request

WRITE

aws s3 cp localfile s3://[bucketname]/test-upload.txt --no-sign-request

READ_ACP

aws s3api get-bucket-acl --bucket [bucketname] --no-sign

WRITE_ACP

aws s3api put-bucket-acl --bucket [bucketname] [ACLPERMISSIONS] --no-sign-request

Elastic Compute Cloud (EC2)

... in other words, Virtual Machines on Amazon.

**Interesting features:**

1) It can be combined with other AWS Services such as S3 to save snapshots.

2) There’s a specific end-point [http://169.254.169.254](http://169.254.169.254) that can be queried to get juicy info about the EC2 instance.
Publicly accessible EC2 snapshots

This can be real fun ... for everyone.

People take snapshots of their EC2 instance and leave them publicly accessible.

All you need is an EC2 instance and you can mount any EC2 snapshot in it.
# Metadata leakage

Having access to [http://169.254.169.254](http://169.254.169.254) via the EC2 instance allows to have access to many juicy info.

<table>
<thead>
<tr>
<th>URL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://169.254.169.254/latest/meta-data/iam/security-credentials/">http://169.254.169.254/latest/meta-data/iam/security-credentials/</a></td>
<td>If there is an IAM role associated it returns its name (which can be used in the next handler).</td>
</tr>
<tr>
<td><a href="http://169.254.169.254/latest/meta-data/iam/security-credentials/role-name">http://169.254.169.254/latest/meta-data/iam/security-credentials/role-name</a></td>
<td>If there is an IAM role associated with the instance, role-name is the name of the role, and role-name contains the temporary security credentials associated with the role (for more information, see Retrieving Security Credentials from Instance Metadata). Otherwise, not present.</td>
</tr>
<tr>
<td><a href="http://169.254.169.254/latest/user-data">http://169.254.169.254/latest/user-data</a></td>
<td>Returns a user-defined script which is run every time a new EC2 instance is launched for the first time.</td>
</tr>
</tbody>
</table>
Nice … but how?

• Proxy feature
• Server-Side Request Forgery
• DNS Rebinding
• …any other way to make a request through an EC2

CloudFront

Content Delivery Network (CDN) helps deliver things faster.
CloudFront

1) request X

2) if X is not in cache, request it from the origin

4) request X from origin

6) resource X is stored in cache for next time

5) X

Edge location

Origin location
A CDN needs a way to uniquely identify a request. Some parts of the HTTP request are used for such unique identification. If the cache keys change, ask new version, otherwise serve a cached version.

```
GET /en?cb=1 HTTP/1.1
Host: www.redhat.com
X-Forwarded-Host: canary
HTTP/1.1 200 OK
Cache-Control: public, no-cache
...
<meta property="og:image"
content="https://canary/cms/social.png" />
```
Practical Web Cache Poisoning
(by James Kettle)

GET /en?cb=1 HTTP/1.1
Host: www.redhat.com
X-Forwarded-Host: a."<script>alert(1)</script>
HTTP/1.1 200 OK
Cache-Control: public, no-cache
...
<meta property="og:image"
content="https://a."><script>alert(1)</script>/cms/social.png" />

X-Forwarded-Host is not a cache key.
If I manage to force CloudFront to save this request, other users will get XSSed.

https://portswigger.net/blog/practical-web-cache-poisoning
AWS Cognito provides developers with an authentication, authorization and user management system that can be implemented in web applications.

Identity pools as they provide access to other AWS services that we might be able to mess with.
Identity ID: us-east-2:ddeb887a-e235-41a1-be75-2a5f675e0944
Request ID: cb3d99ba-b2b0-11e8-9529-0b4be486f793
SecretKey: wJE/[REDACTED]Kru76jp4i
AccessKey ID: ASI[REDACTED]MAO3
SessionToken AgoGb3JpZ21uELf[REDACTED]wWeDg8CjW9MPerytwF

Buckets:

mindeds3log
mindeds3test01

Link?? Coming soon...
Conclusione

WITH GREAT POWER COMES GREAT RESPONSIBILITY.

OWASP
Open Web Application Security Project
That's all Folks!