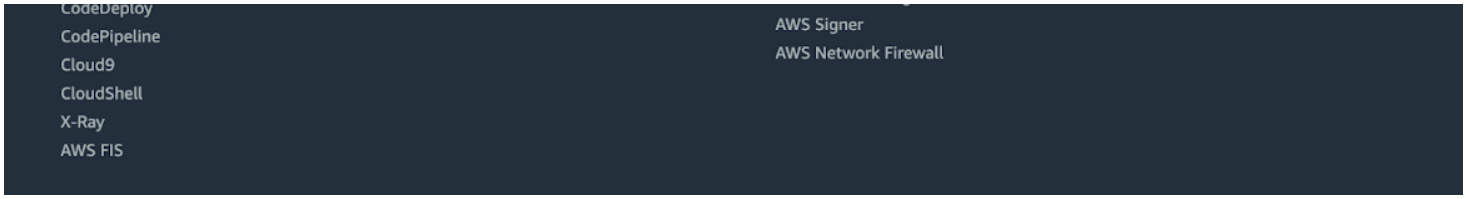
Amazon AWS

Amazon AWS is a public cloud service provider. As of their most recent financial disclosures, AWS accounts for the bulk of Amazon's profit. Most major enterprises leverage AWS in some form or another for Compute Services, Big Data or Machine Learning, Data Archive, Video Streaming, IoT, etc. The number of services AWS supports is so vast that we can barely fit it all in this screenshot.



Your eyes don't deceive you. You can access robots, blockchain, satellites, and quantum computing from AWS.

AWS divides its infrastructure into Regions, mostly independent clusters of datacenters. Within each region are availability zones (AZ). Each AZ in a region leverages separate power grids and usually are located in different flood plains. This redundancy allows you to establish highly resilient architectures to withstand significant weather or geological events, or more frequently, hardware or facility failures.

Because regions are independent - you'll get different answers to questions depending on the region you are querying. You can specify a region with the `--region` option to the AWS CLI.

You can access AWS via the AWS Console, AWS CLI, AWS API, or the associated SDKs for your favorite programming languages.

Amazon S3

Amazon S3 (Simple Storage Service) is their hosted object storage service. Objects are stored in Buckets. To highly simplify the concept of object storage, Buckets are key-value stores, with the Object Key being a full pathname for a file and the value being the contents of the file. S3 is a publicly hosted service - it doesn't exist behind a corporate firewall, making it convenient for hosting public content. AWS has an entire feature set around hosting a public website in S3.

AWS Buckets use a global namespace. Only one AWS customer can create a bucket named `bestfestivalcompany-images` .

Amazon S3 is used for more than public hosting. It has many uses for data archive, video processing, regulatory record retention, etc. The challenge for Best Festival Company, like any enterprise using S3, is that sometimes data gets mixed up, and data that shouldn't be public gets made public.

Discovering Bucket Names

There are many ways to discover the names of Buckets. One of the easiest ways is when a company embeds content hosted in S3 on their website. Images, PDFs, etc., can all be hosted cheaply in S3 and linked from another site. These links will look like this:

```
http://BUCKETNAME.s3.amazonaws.com/FILENAME.ext
```

or

```
http://s3.amazonaws.com/BUCKETNAME/FILENAME.ext
```

In both these cases, it is easy to identify the name of the S3 bucket. Now, what can we do with that information?

Listing the Contents of Buckets

Amazon S3 is one of AWS's oldest services. It's so old that it has two different methods of access control: Bucket Policies and S3 ACLs. This leads to great confusion for developers who must manage policies, ACLs, and the differences between Any User and Authenticated Users.

Many buckets that contain public information allow you to list the contents of the bucket. In your AttackBox, try running the command:

```
curl http://irs-form-990.s3.amazonaws.com/
```

That massive pile of XML is a listing of all the IRS Form 990 filings for US Tax-Exempt corporations. AWS makes this data available as a public dataset.

If mentally parsing XML that contains no line breaks isn't your cup of tea, the AWS CLI also provides the ability to list the contents of a bucket (You probably want to hit Ctrl-C after a few seconds, there are a lot of US non-profit organizations).

```
aws s3 ls s3://irs-form-990/ --no-sign-request
```

The option `--no-sign-request` allows you to request data from S3 without being an AWS Customer.

Downloading Objects

Downloading an object from S3 is also easy. You can use curl:

```
curl http://irs-form-990.s3.amazonaws.com/201101319349101615_public.xml
```

or the AWS CLI:

```
aws s3 cp s3://irs-form-990/201101319349101615_public.xml . --no-sign-request
```

Note the two different URIs for an object. Objects can be addressed with http:// or via s3://

The different levels of Amazon S3 Authentication

In Amazon S3, Object permissions are different from Bucket permissions. Bucket permissions allow you to list the objects in a bucket, while the object's permissions will enable you to download the object. In the case of the irs-form-990 bucket, both the bucket and all the objects in the bucket are publicly readable. But that doesn't have to be the case. Objects can be readable while the bucket is not, or the bucket can be publicly readable, but the Objects are not.

Note: you can also have public write permissions to a Bucket. This is generally a bad idea and has been the vector of several crypto-mining incidents.

There are also two levels of public buckets and objects. The first level is "Anyone." This is what you experienced with the irs-form-990 bucket. You could just hit that URL from your local browser. The second level is just as public - and that is public to Any AWS Customer (what AWS foolishly called AuthenticatedUsers for many years). Anyone with a credit card can create an AWS account; therefore, Authenticated Users doesn't provide much data protection.

ACL Name	BUCKET	OBJECT
Anyone	Anonymously list contents of the bucket via curl or with `aws s3 ls --no-sign-request`	Ability to download via curl or `aws s3 cp --no-sign-request`
AuthenticatedUsers	Can only list the bucket with active AWS keys via `aws s3 ls`	You can only download the object with active AWS Keys via `aws s3 cp`

AWS IAM

Excluding a few older services like Amazon S3, all requests to AWS services must be signed. This is typically done behind the scenes by the AWS CLI or the various Software development Kits that AWS provides. The signing process leverages IAM Access Keys. These access keys are one of the primary ways an AWS account is compromised.

IAM Access Keys

IAM Access Keys consist of an Access Key ID and the Secret Access Key.

Access Key IDs always begin with the letters AKIA and are 20 characters long. These act as a user name for the AWS API. The Secret Access Key is 40 characters long. AWS generates both strings; however, AWS doesn't make the Secret Access Key available to download after the initial generation.

There is another type of credentials, short-term credentials, where the Access Key ID begins with the letters ASIA and includes an additional string called the Session Token.

Conducting Reconnaissance with IAM

When you find credentials to AWS, you can add them to your AWS Profile in the AWS CLI. For this, you use the command:

```
aws configure --profile PROFILENAME
```

This command will add entries to the `.aws/config` and `.aws/credentials` files in your user's home directory.

Once you have configured a new profile with the new access keys, you can execute any command using this other set of credentials. For example, to list all the S3 Buckets in the AWS account you have found credentials for, try:

```
aws s3 ls --profile PROFILENAME
```

ProTip: Never store a set of access keys in the [default] profile. Doing so forces you always to specify a profile and never accidentally run a command against an account you don't intend to.

A few other common AWS reconnaissance techniques are:

1. Finding the Account ID belonging to an access key:
   ```
   aws sts get-access-key-info --access-key-id AKIAEXAMPLE
   ```
2. Determining the Username the access key you're using belongs to
   ```
   aws sts get-caller-identity --profile PROFILENAME
   ```
3. Listing all the EC2 instances running in an account
   ```
   aws ec2 describe-instances --output text --profile PROFILENAME
   ```
4. Listing all the EC2 instances running in an account in a different region
   ```
   aws ec2 describe-instances --output text --region us-east-1 --profile PROFILENAME
   ```

AWS ARNs

An Amazon ARN is their way of generating a unique identifier for all resources in the AWS Cloud. It consists of multiple strings separated by colons. The format is:

```
arn:aws:<service>:<region>:<account_id>:<resource_type>/<resource_name>
```