Hacking with AWS: incorporating leaky buckets into your OSINT workflow

By Vasilios Hioureas

Published: 2019-09-12 · Archived: 2026-04-05 23:17:07 UTC

Penetration testing is often conducted by security researchers to help organizations identify holes in their security and fix them, before cybercriminals have the chance. While there's no malicious intent for the researcher, part of his job is to think and act like a cybercriminal would when hacking, or attempting to breach, an enterprise network.

Therefore, in this article, I will review Amazon AWS buckets as an avenue for successful penetration tests. The case study I'm using is from a reconnaissance engagement I conducted against a business. I will specifically focus on how I was able to use AWS buckets as an additional avenue to enrich my results and obtain more valuable data during this phase.

NOTE: For the safety of the company, I will not be using real names, domains, or files obtained. However, the concept will be clearly illustrated despite the lack of specifics.

The goal of this article is to present an alternative or an additional method for professionals conducting pen-tests against an organization. In addition, I hope that it may also serve as a warning for companies deciding to use AWS to host private data and a reminder to secure potentially leaky buckets.

What is an AWS bucket?

Amazon Simple Storage Service (S3) provides an individual or business the ability to store and access content from Amazon's cloud. This concept is not new, however, because businesses use AWS buckets to not only store and share files between employees, but also host Internet-facing services, we have seen a wealth of private data being exposed publicly.

The types of data we have discovered range from server backups and backend web scripts to company documents and contracts. Files within S3 are organized into "buckets," which are named logical containers accessible by a static URL.

A bucket is typically considered public if any user can list the contents of the bucket, and private if the bucket's contents can only be listed or written by certain S3 users.

Checking if a bucket is public or private is easy. All buckets have a predictable and publicly accessible URL. By default this URL will be either of the following:

s3.amazonaws.com/[bucket_name]/
or
[bucket_name].s3.amazonaws.com/

Pen-test workflow: hacking phases

Let's begin by talking about the first phase of a penetration test: reconnoissance. The purpose of this phase is to gather as much information about a target in order to build an organized list of data which will be used in future hacking phases (scanning, enumeration, and gaining access).

In general, some of the data which pen-testers hope to obtain during this phase is as follows:

- Names, email addresses, and phone numbers of employees (to be used for phishing)
- Details of systems used by the business (domains, IPs, and other services to be enumerated in future phases)
- Files containing usernames, passwords, leaked data, or other access-related items
- Spec sheets, contracts, vendor documents, infrastructure outlines, notes
- Customer data (side channel compromise of a trusted third party can be just as valuable as the target themselves)

Alternate infrastructure

All of the items above are examples of things we can and have found while scouring AWS buckets. But first, before we get into the information discovered, let's talk about finding the buckets themselves and the role that an S3 bucket search can play in a pen-test.

An easy first step in any recon phase is to enumerate the primary domain via brute force hacking or any other method, looking to hopefully find subdomains which may be hosting services within a company. This is pretty standard procedure, but a business does not always have all of its data or resources internally hosted. Often there are unofficial IPs that may be hosted offsite serving a secondary role to the primary business or possibly hosting developer resources or even file storage.

While the company's internal services, such as mail, websites, firewalls, security, and documentation may be hosted within a subdomain, for several reasons there are still possibilities that offsite or separate servers are used by the company. For this reason, it is a good idea to expand your search, using not only Google hacks and keywords to look for related services or domains.

External resource example

Once specific example related to this was a PACS server data leak from one of UCLA's medical centers. Now while this server was technically operated by a UCLA med centers, it was not an official service of UCLA, so to speak.

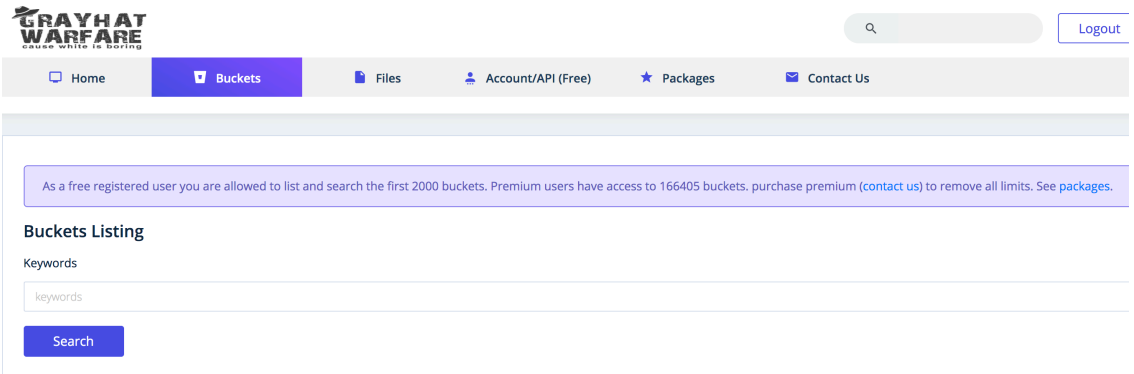
Translation: This server was not part of the UCLA domain. It happened to be independently hosted by one of the residing med center employees, yet in a more obscure way, it was still related to the company. This is an example of the sort of side channel opportunities available to criminals.

Finding leaky buckets

Moving forward, an Amazon S3 bucket is a prime example of one such "unrelated" service not directly tied to the business' infrastructure. My main purpose for introducing this is to give pen-testers a new hacking avenue in addition to Google hacks. Because although Google searching on a company can lead you to their AWS bucket, it is more effective to search the open buckets directly.

There are a number of tools that can be used to discover wide-open buckets. The one I'd like to highlight is the web application Grayhat Warfare. Of all the tools I use, it is the most user friendly and most easily accessible.

As you can see below, it is quite intuitive:



Let's take a look at this application and see how a pen-tester might try to use it to discover a bucket owned by an organization.

There are a few ways in which pen testers can uncover unsecured data at their companies. One is by searching for filenames that you might expect to find from the target organization. For example, knowing some of the services or products the enterprise produces, you might search those specific product names, or *company_name*.bak.

Additionally, having completed some other recon, incorporating usernames in this search can lead to some results. In general, this is the part of the process that requires all of the creativity and thinking outside of the box.

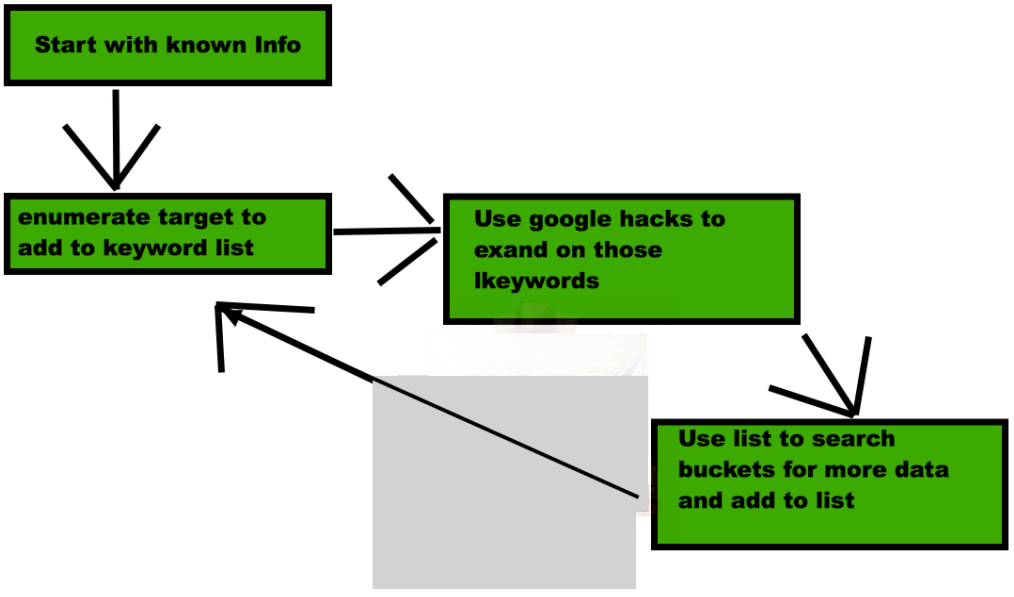
Hacking with AWS case study

Now let's dig into the case study to see these recon methods in action. In this specific case, the target was an entertainment company that produces content for the music industry. From some Google searching, I happened to come across the fact that they developed an Android app. The app name in this case had no relation to the actual company name; these are exactly the discoveries needed to make to expand searches of leaky buckets.

Using the name of the app and searching it within Grayhat Warfare, I was lucky enough to find an AWS bucket containing a file with the name of the app. One important thing to note is that the bucket name and URL were also completely different and unrelated to the name of the company.

This unrelated naming scheme is often by design. Rather than creating an obvious name, business infrastructure architects often name servers according to themes. You might see planets, Greek gods, Star Wars characters, or favorite bands. This makes searching for services a bit more obscure for a hacker.

Once the app filename was found within a bucket, it was simply a matter of manually looking through the rest of the files to verify if the bucket in fact belonged to the target organization. This eventuality led me to find even more info to use for a deeper recon search.



One amazing find on this server was actually a zip file with the app's source code. This contained database IPs, usernames, and passwords. This is something that may never have been discovered using traditional recon methods since the IP happened to be an offsite DreamHost account. It was completely untied to any of the company's resources.

OSINT standards plus AWS buckets = more data

The main point I wanted to illustrate from my test case is how hacking with AWS can be incorporated into the pen-test workflow as an iterative fingerprinting cycle. Using Google hacks, Shodan, and social networks are a standard for open source intelligence (OSINT). We use these traditional methods to gather as much data as possible, then once we have found as much as we can find, we can blast that data against bucket search tools to retrieve deeper info.

From that point, pen-testers can restart the whole search process with this new data. Recon can often be recursive, one result leading to another, leading to another. However, incorporating AWS bucket searching into your pen-test workflow can provide data that may not have been obtained using the other methods.

If any readers have other hacking or search tools they have come across or alternative methods for recon, please feel free to mention them below in the comments.

About the author

Reverse engineer, software developer, malware analyst, smart city hacker, RF hacker, IOT exploit researcher.
