iD

I use Zip Bombs to Protect my Server

Bots be warned

By **Ibrahim Diallo**

Published Apr 17 2025 ~ 5 minutes read

The majority of the traffic on the web is from bots. For the most part, these bots are used to discover new content. These are RSS Feed readers, search engines crawling your content, or nowadays AI bots crawling content to power LLMs. But then there are the malicious bots. These are from spammers, content scrapers or hackers. At my old employer, a bot discovered a wordpress vulnerability and inserted a malicious script into our server. It then turned the machine into a botnet used for DDOS. One of my first websites was yanked off of Google search entirely due to bots generating spam. At some point, I had to find a way to protect myself from these bots. That's when I started using zip bombs.

A zip bomb is a relatively small compressed file that can expand into a very large file that can overwhelm a machine.

A feature that was developed early on the web was compression with gzip. The Internet being slow and information being dense, the idea was to compress data as small as possible before transmitting it through the wire. So an 50 KB HTML file, composed of text, can be compressed to 10K, thus saving you 40KB in transmission. On dial up Internet, this meant downloading the page in 3 seconds instead of 12 seconds.

This same compression can be used to serve CSS, Javascript, or even images. Gzip is fast, simple and drastically improves the browsing experience. When a browser makes a web request, it includes the headers that signals the target server that it can support compression. And if the server also supports it, it will return a compressed version of the expected data.

```
Accept-Encoding: gzip, deflate
```

Bots that crawl the web also support this feature. Especially since their job is to ingest data from all over the web, they maximize their bandwidth by using compression. And we can take full advantage of this feature.

1MB to 10MB file which they are happy to ingest. For the most part, when they do, I never hear from them again. Why? Well, that's because they crash right after ingesting the file.

```
Content-Encoding: deflate, gzip
```

What happens is, they receive the file, read the header that instructs them that it is a compressed file. So they try to decompress the 1MB file to find whatever content they are looking for. But the file expands, and expands, and expands, until they run out of memory and their server crashes. The 1MB file decompresses into a 1GB. This is more than enough to break most bots. However, for those pesky scripts that won't stop, I serve them the 10MB file. This one decompresses into 10GB and instantly kills the script.

Before I tell you how to create a zip bomb, I do have to warn you that you can potentially crash and destroy your own device. Continue at your own risk. So here is how we create the zip bomb:

```
dd if=/dev/zero bs=1G count=10 | gzip -c > 10GB.gz
```

Here is what the command does:

1. `dd` : The dd command is used to copy or convert data.
2. `if` : Input file, specifies `/dev/zero` a special file that produces an infinite stream of zero bytes.
3. `bs` : block size, sets the block size to 1 gigabyte (1G), meaning dd will read and write data in chunks of 1 GB at a time.
4. `count=10` : This tells dd to process 10 blocks, each 1 GB in size. So, this will generate 10 GB of zeroed data.

We then pass the output of the command to gzip which will compress the output into the file 10GB.gz. The resulting file is 10MB in this case.

On my server, I've added a middleware that checks if the current request is malicious or not. I have a list of black-listed ips that try to scan the whole website repeatedly. I have other heuristics in place to detect spammers. A lot of spammers attempt to spam a page, then come back to see if the spam has made it to the page. I use this pattern to detect them. It looks something like this:

```
if (ipIsBlackListed() || isMalicious()) {
    header("Content-Encoding: deflate, gzip");
    header("Content-Length: "+ filesize(ZIP_BOMB_FILE_10G)); // 10 MB
    readfile(ZIP_BOMB_FILE_10G);
    exit;
}
```

One more thing, a zip bomb is not foolproof. It can be easily detected and circumvented. You could partially read the content after all. But for unsophisticated bots that are blindly crawling the web disrupting servers, this is a good enough tool for protecting your server.

You can see it in action in this replay of my server logs.

Did you like this article? You can buy me a coffee.
Share your insightful comments here.

Previous: Part 8: Deployment Planning & Infrastructure Setup

Next: Part 9: Launch & Post-Deployment Monitoring

On a related note, here are some interesting articles.

Track page read not page views

When I first started to work on the web, I

The best way to add custom web fonts

Changing Apache Server Signature

A quick question. Do you need to know

iD

Comments

There are no comments added yet.

Let's hear your thoughts

Enter your insightful comments here ...

Your Name (Required)		
Your Email (Required)		For my eyes only
Your Website		

Would you like to sign up to the news letter? ☐ **← Click here**

Post Comment

About Me

First of all, Wow! You scrolled all the way down. That means you want to know more about me. Well, here is a summary of who I am and what I do. I started this blog because ... wait I have a link for that too.

Hey! Have you heard about humans.txt? Well I kinda liked the idea and did my own version.

You can find me on:

Twitter YouTube Soundcloud

Don't hesitate to say hi, it's what keeps me going :)

Designed by Yours truly.