Vocational English II (Mesleki Yabancı Dil II) Week 11

Engineering Faculty

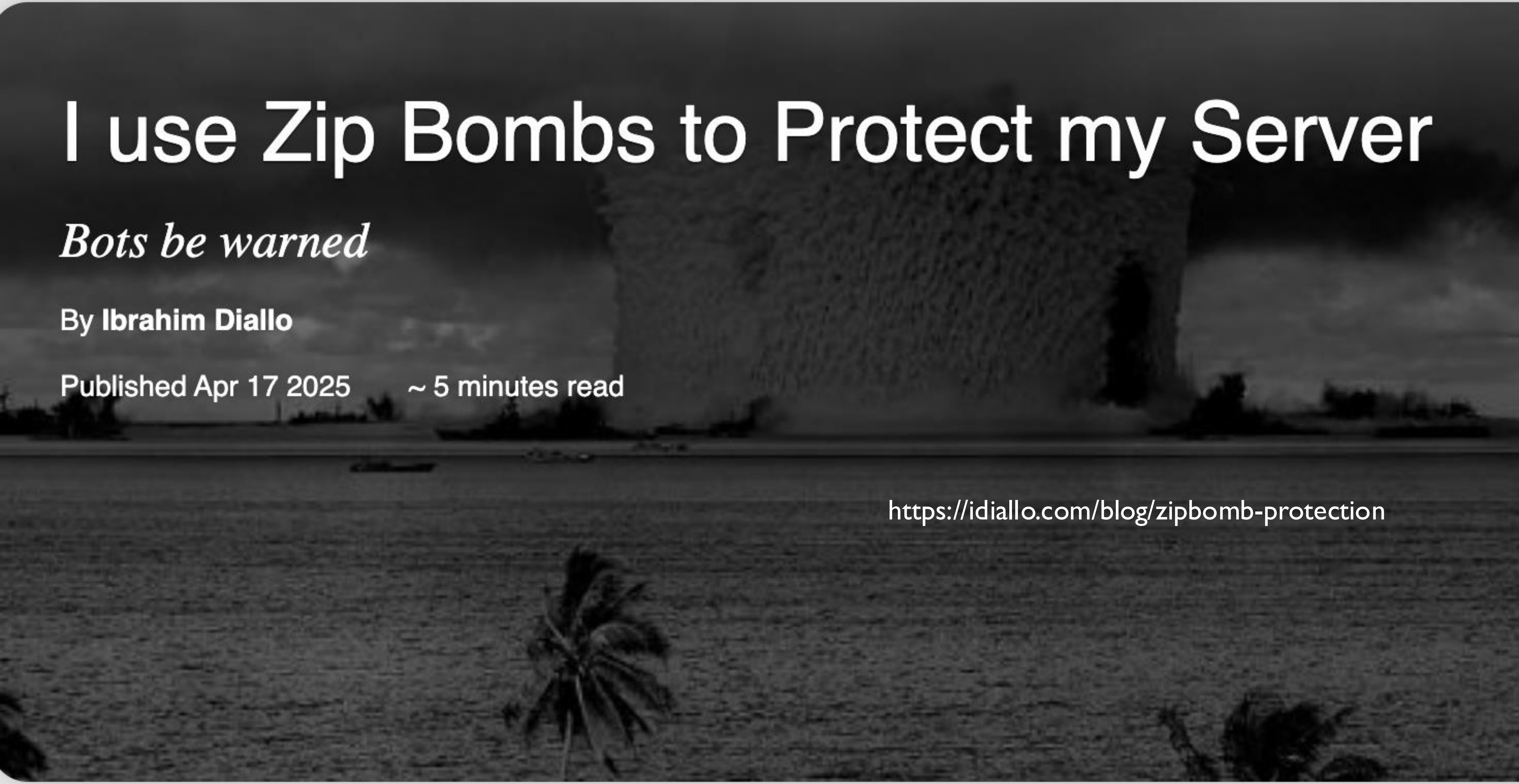
Computeer Engineering

Prepared by: Dr Ercan Ezin

INTRODUCTION

THIS WEEK WE WILL WORK ON

Cyber Security



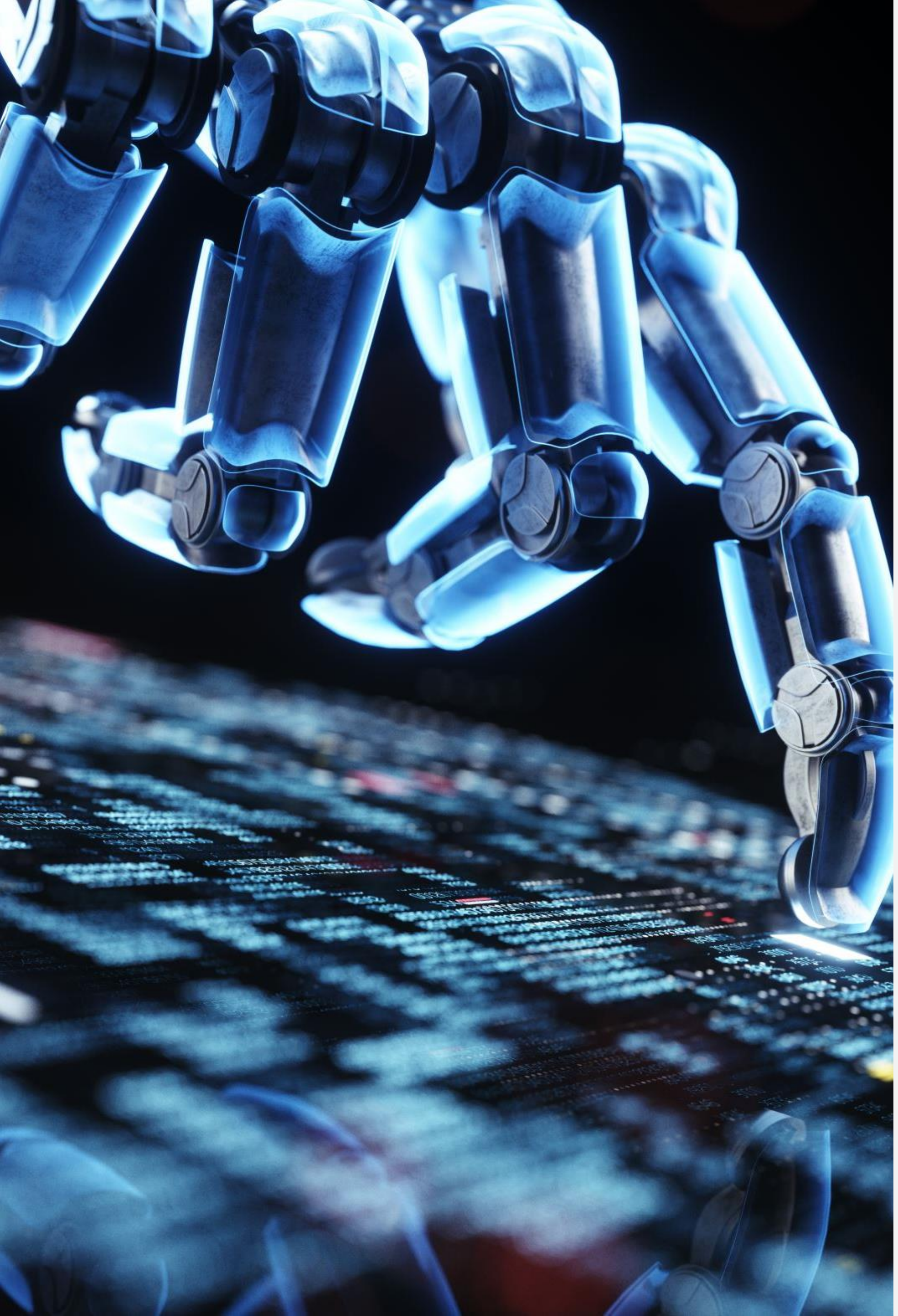
I use Zip Bombs to Protect my Server

Bots be warned

By **Ibrahim Diallo**

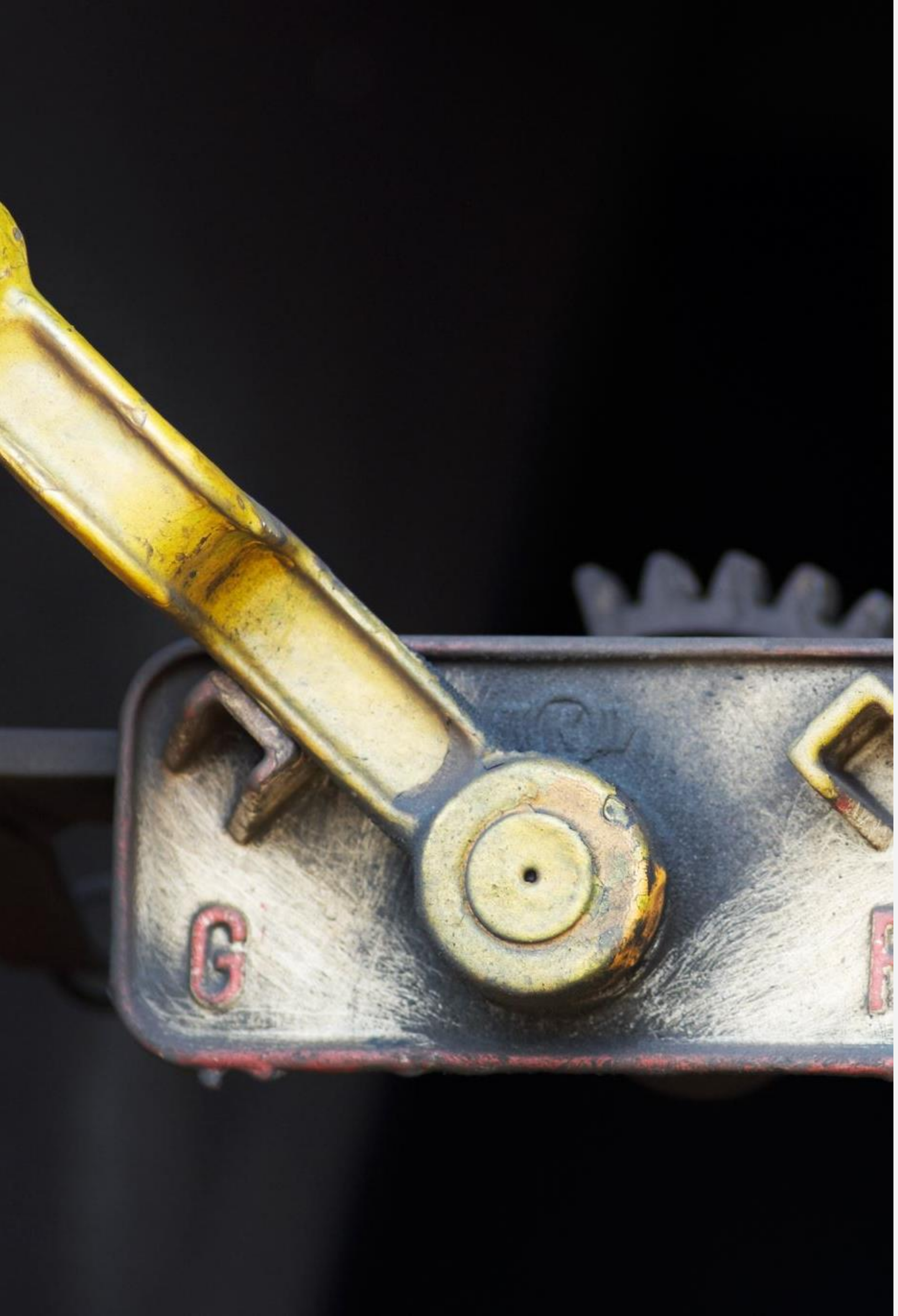
Published Apr 17 2025 ~ 5 minutes read

https://idiallo.com/blog/zipbomb-protection



THE WEB IS CRAWLING WITH BOTS

- 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.



WHAT IS A ZIP BOMB?

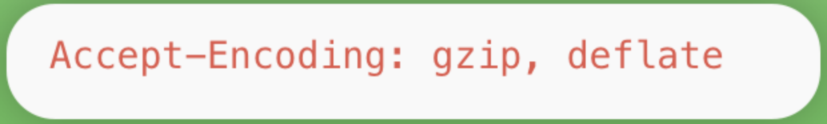
- A zip bomb is a relatively small **compressed** file that can expand into a very large file that can **overwhelm** a machine.

THE POWER OF COMPRESSION

- 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, a 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**.

HOW BOTS USE COMPRESSION

When a browser makes a web request, it includes the **headers** that signal 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.



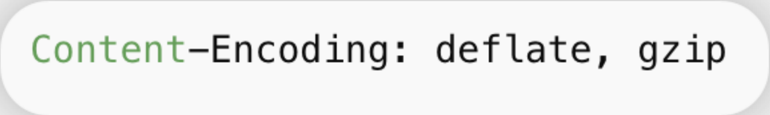
```
Accept-Encoding: gzip, deflate
```

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**.

TURNING COMPRESSION AGAINST BOTS

- On this blog, I **often** get bots that scan for security vulnerabilities, which I ignore for the most part. But when I detect that they are either trying to **inject** malicious attacks, or are **probing** for a response, I return a 200 OK response, and **serve them** a gzip response.

- I vary from a 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.



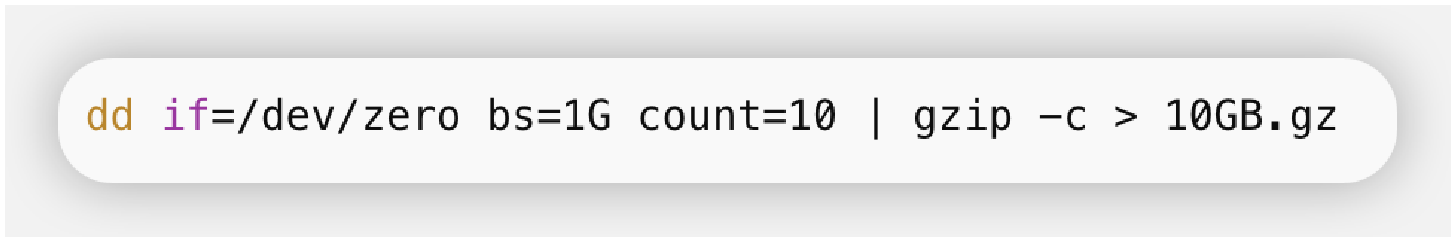
```
Content-Encoding: deflate, gzip
```

CRASHING THE BOTS

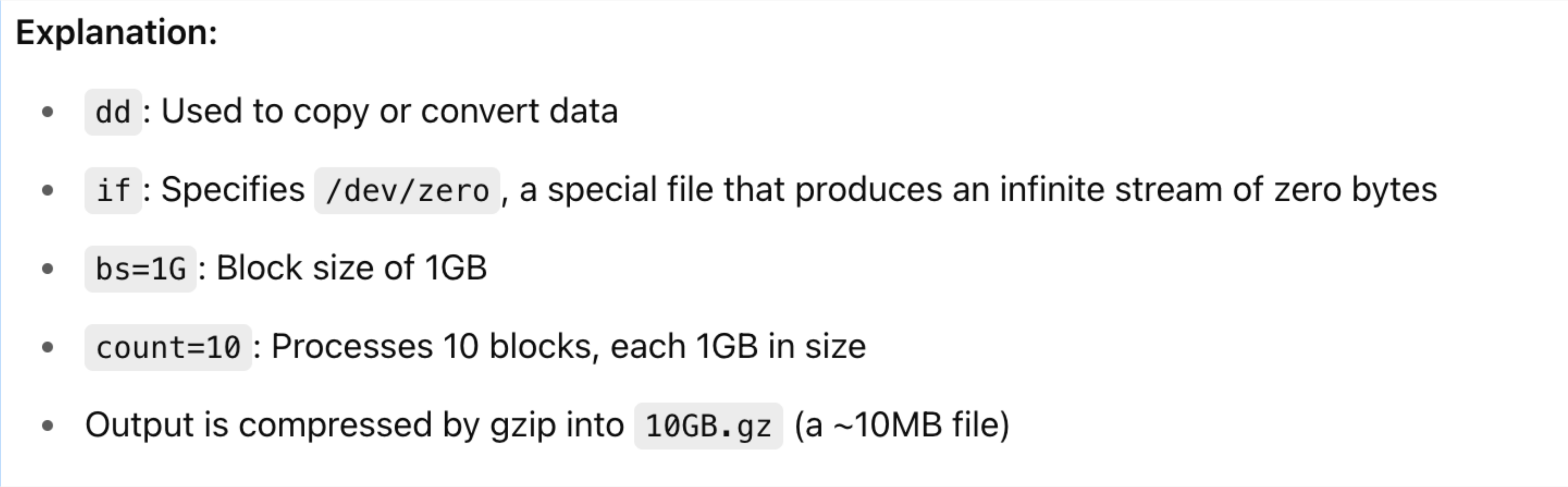
- **What happens is**, they receive the file, read the header that instructs them that it is a compressed file. So they try to **decompress** the 1MB file to find whatever content they are looking for.
  But the file expands, and **expands, and expands**, until they **run out of memory** and their server 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.

HOW TO CREATE A ZIP BOMB

- 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
```

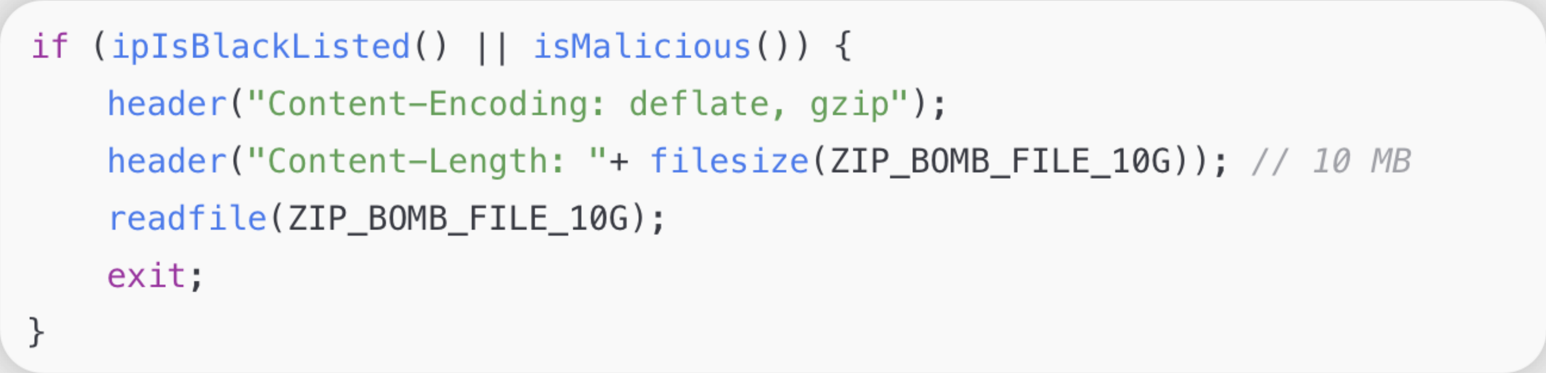


Explanation:

- `dd` : Used to copy or convert data
- `if` : Specifies `/dev/zero` , a special file that produces an infinite stream of zero bytes
- `bs=1G` : Block size of 1GB
- `count=10` : Processes 10 blocks, each 1GB in size
- Output is compressed by gzip into `10GB.gz` (a ~10MB file)

DEPLOYING ZIP BOMBS STRATEGICALLY

- 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;
}
```

The only price I pay is serving a 10MB file occasionally. If I have an article going viral, I decrease it to the 1MB file.
Note: A zip bomb is not **foolproof**. But for **unsophisticated** bots, it's good enough.

LISTENING ACTIVITY

https://www.youtube.com/watch?v=fKuqYQdqRIs

How To Protect Your Linux Server From Hackers!

WORDS OF THE WEEK

1. **Bot** An automated script or program that performs tasks such as crawling, spamming, or hacking.
2. **Crawler** A bot used by search engines or AI to browse and index web content.
3. **Spammer** A bot or user that floods websites with unwanted content or links.
4. **Hacker** A person or bot that exploits vulnerabilities in software or systems.
5. **Botnet** A network of compromised machines controlled to perform coordinated cyberattacks.
6. **DDoS** Distributed Denial of Service attack, where many machines flood a server to crash it.
7. **Zip Bomb** A compressed file that decompresses into a massive file to overwhelm and crash systems.
8. **Compression** Reducing the size of data for faster transmission or storage.
9. **Decompression** Expanding compressed data back to its original or usable form.
10. **gzip** A popular compression utility and format used on the web to minimize data size.
11. **Header** Metadata sent in web requests/responses, often used for controlling content and behavior.
12. **Content-Encoding** An HTTP header specifying how content is compressed (e.g., gzip, deflate).
13. **Middleware** Software layer that handles requests before they reach the application logic.
14. **Blacklisting** Blocking certain IPs or users based on predefined criteria, such as malicious behavior.
15. **Heuristic** A rule-based method for identifying patterns, often used in spam or threat detection.
16. **Vulnerability** A weakness in software or systems that can be exploited by attackers.
17. **Script** A small program or automated sequence of actions often used by bots or attackers.
18. **Deflate** A compression algorithm similar to gzip, used in web communications.
19. **Bandwidth** The data transfer capacity of a network, often optimized using compression.
20. **RSS (Really Simple Syndication)** A web feed format used to publish frequently updated information like blog posts or news articles.

EOF*

*End of Fun/File