I Came to Drop Bombs
Auditing the Compression Algorithm Weapons Cache

Cara Marie
NCC Group
Blackhat USA 2016
About Me

• NCC Group Senior Security Consultant
  Pentested numerous networks, web applications, mobile applications, etc.
• Hackbright Graduate
• Ticket scalper in a previous life
• @bones_codes | cara.marie@nccgroup.com
What is a Decompression Bomb?

A decompression bomb is a file designed to crash or render useless the program or system reading it.
Vulnerable Vectors

- Chat clients
- Image hosting
- Web browsers
- Web servers
- Everyday web-services software
- Everyday client software
- Embedded devices (especially vulnerable due to weak hardware)
- Embedded documents
- Gzip’d log uploads
A History Lesson

early 90’s  •  ARC/LZH/ZIP/RAR bombs were used to DoS FidoNet systems

2002  •  Paul L. Daniels publishes Arbomb (Archive “Bomb” detection utility)

2003  •  Posting by Steve Wray on FullDisclosure about a bzip2 bomb antivirus software DoS

2004  •  AERAsec Network Services and Security publishes research on the various reactions of antivirus software against decompression bombs, includes a comparison chart

2014  •  Several CVEs for PIL are issued – first release July 2010 (CVE-2014-3589, CVE-2014-3598, CVE-2014-9601)

2015  •  CVE for libpng – first release Aug 2004 (CVE-2015-8126)
Why Are We Still Talking About This?!?
Why Are We Still Talking About This?!?

A zip **bomb**, also known as a zip of death or **decompression bomb**, is a malicious archive file designed to crash or render useless the program or system reading it. It is often employed to disable antivirus software, in order to create an opening for more traditional viruses.

**Zip bomb - Wikipedia, the free encyclopedia**

A zip bomb, also known as a zip of death or decompression bomb, is a malicious archive file designed to crash or render useless the program or system reading it. It is often employed to disable antivirus software, in order to create an opening for more traditional viruses.

Billion laughs · Logic bomb · Busy beaver · Email bomb
Compression is the New Hotness
Who This Is For
Who This Is For
An archive bomb, a.k.a. zip bomb, is often employed to disable antivirus software, in order to create an opening for more traditional viruses

- Singly compressed large file
- Self-reproducing compressed files, i.e. Russ Cox’s Zips All The Way Down
- Nested compressed files, i.e. 42.zip
42.zip (42.374B) is comprised of:

16 x 4294967295 = 68.719.476.720 (68GB)
16 x 68719476720 = 1.099.511.627.520 (1TB)
16 x 1099511627520 = 17.592.186.040.320 (17TB)
16 x 17592186040320 = 281.474.976.645.120 (281TB)
16 x 281474976645120 = 4.503.599.626.321.920 (4.5PB)

– Each containing a single 4.3GB file –
Compression Bombs

Ratio Calculation

Compression Ratio = \frac{\text{Uncompressed Content}}{\text{Compressed Content}}

1048576 = \frac{10485760\text{KB (10GB)}}{10\text{KB}}
The Archives
Compression Ratio Graph
The Archives
Compression Ratio Graph (sans bzip2)
## The Archives

### Compression Ratios

<table>
<thead>
<tr>
<th>Utility</th>
<th>Size</th>
<th>Compression Ratio</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>bzip2</td>
<td>7KB</td>
<td>~1427411:1</td>
<td>Burrows-Wheeler</td>
</tr>
<tr>
<td>xar (bzip2)</td>
<td>9KB</td>
<td>~1198921:1</td>
<td>Burrows-Wheeler</td>
</tr>
<tr>
<td>7z (gzip)</td>
<td>1.5MB</td>
<td>~6848:1</td>
<td>DEFLATE</td>
</tr>
<tr>
<td>xz</td>
<td>1.5MB</td>
<td>~6875:1</td>
<td>LZMA</td>
</tr>
<tr>
<td>RAR</td>
<td>5.2MB</td>
<td>~2003:1</td>
<td>LZSS/PPM</td>
</tr>
<tr>
<td>LZFSE</td>
<td>6.3MB</td>
<td>~1625:1</td>
<td></td>
</tr>
<tr>
<td>gzip</td>
<td>10.2MB</td>
<td>~1029:1</td>
<td>DEFLATE</td>
</tr>
<tr>
<td>ZIP</td>
<td>10.2MB</td>
<td>~1029:1</td>
<td>DEFLATE</td>
</tr>
<tr>
<td>xar (default)</td>
<td>10.2MB</td>
<td>~1028:1</td>
<td>DEFLATE</td>
</tr>
<tr>
<td>LZ4</td>
<td>41.2MB</td>
<td>~258:1</td>
<td>LZ77</td>
</tr>
</tbody>
</table>

Ratios calculated from a zero-generated 10GB file
Mitigations

Security 101

- Never rely on client-side checks for security
- Perform server-side checks to validate:
  - File format is expected for context
  - File size will not exceed maximum limit
  - File name is sane/safe
  - File names are validated to avoid symlink/hardlink or directory traversal attacks
Mitigations

The Archives

Limit the amount of resources available to the process and its children

• For Linux platforms, cgroups can and should be used to limit both CPU and memory usage

• In Python resource limits can be configured via the `resource` module’s `setrlimit` and `RLIMIT*` directives:

```python
import resource
rsrc = resource.RLIMIT_DATA
resource.setrlimit(rsrc, (1024000, hard)) # limit to 1MB
```

• Ruby’s `Process` module has similar `RLIMIT` directives
Restrict output file size and number of extracted files, and throw an exception if either of these limits are reached

```python
import zlib

def decompress(data, maxsize=1024000):
    dec = zlib.decompressobj()
    data = dec.decompress(data, maxsize)
    if dec.unconsumed_tail:
        raise ValueError("Possible bomb")
    del dec
    return data
```
Images can be highly effective in causing a denial of service for:

- Web servers and clients
- Mobile clients
Not Just a Pretty Picture

Compression Ratio Graph
Not Just a Pretty Picture

Compression Ratio Graph (the Universals)
## Not Just a Pretty Picture

### Compression Ratios

<table>
<thead>
<tr>
<th>Format</th>
<th>Size</th>
<th>Compression Ratio</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>JPEG 2000</td>
<td>552B</td>
<td>~181159:1</td>
<td>DWT</td>
</tr>
<tr>
<td>WebP*</td>
<td>4KB</td>
<td>~24414:1</td>
<td>LZ77</td>
</tr>
<tr>
<td>ZopfliPNG</td>
<td>12KB</td>
<td>~8138:1</td>
<td>WebP</td>
</tr>
<tr>
<td>GIF</td>
<td>68KB</td>
<td>~1436:1</td>
<td>LZW</td>
</tr>
<tr>
<td>WebP</td>
<td>177KB</td>
<td>~552:1</td>
<td>LZ77</td>
</tr>
<tr>
<td>TIF</td>
<td>292KB</td>
<td>~334:1</td>
<td>LZW</td>
</tr>
<tr>
<td>PNG</td>
<td>316KB</td>
<td>~309:1</td>
<td>DEFLATE</td>
</tr>
<tr>
<td>JPEG</td>
<td>586KB</td>
<td>~167:1</td>
<td>DCT</td>
</tr>
</tbody>
</table>

Ratios calculated from 10Kx10K, 8-bit single-color img (~95MB)

- WebP restricts image input to a maximum of 16383 pixels
* Initial WebP entry is the Imagemagick implementation
Programmatically check image dimensions prior to processing

- libpng allows size limitations to be placed using
  `png_set_user_limits()`
  (the default is 1,000,000 by 1,000,000 pixels)
- For Python, this can be done using PIL’s `Image` module:

```python
from PIL import Image
im = Image.open(image_filename)
width, height = im.size
# Check image dimensions
if (width < MAX_IMAGE_WIDTH) and (height < MAX_IMAGE_HEIGHT):
    # do stuff
```
Mitigations
Not Just a Pretty Picture

- Use workers to perform process intensive tasks
- Limit the amount of resources available to the process and its children
  - libpng allows users to impose memory consumption and ancillary chunk limits via `png_set_chunk_malloc_max()` and `png_set_chunk_cache_max()`
- Contents are scrubbed to minimum required (exif data, avoiding image based XSS, etc.)
HTTP bombs can be used to target:

- Web servers
- Web clients (includes mobile)
- Embedded devices
Bombs in Flight
Compression Ratio Graph
## Bombs in Flight

### Compression Ratios

<table>
<thead>
<tr>
<th>Format</th>
<th>Size</th>
<th>Compression Ratio</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gzip</td>
<td>10.2MB</td>
<td>~1029:1</td>
<td>DEFLATE</td>
</tr>
<tr>
<td>Zopfli</td>
<td>10.3MB</td>
<td>~1017:1</td>
<td>DEFLATE</td>
</tr>
<tr>
<td>bzip2</td>
<td>7KB</td>
<td>~1427410:1</td>
<td>Burrows–Wheeler</td>
</tr>
<tr>
<td>brotli</td>
<td>27KB</td>
<td>~387844:1</td>
<td>LZ77</td>
</tr>
<tr>
<td>LZMA</td>
<td>1.5MB</td>
<td>~7089:1</td>
<td></td>
</tr>
</tbody>
</table>

Ratios calculated from a zero-generated 10GB HTML file

- Zopfli binary restricts content to 2GB
- Bzip2 is supported by Lighttpd
- Brotli is supported in Firefox and Chrome (currently only for HTTPS)
- LZMA is supported in Opera beta 33
Limit the amount of resources available to the process and its children

- For Apache, use the `RLimit*` directives: `RLimitCPU`, `RLimitMEM`, and `RLimitNPROC`
- For Nginx, use the `worker_rlimit_core`, `worker_rlimit_nofile`, and `worker_processes` directives
- For Linux platforms, cgroups can be used to limit both CPU and memory usage
Mitigations

Bombs in Flight

- Limit request sizes
  - This can be done in Apache using the `LimitRequestBody` directive
  - For Nginx use the `client_max_body_size` directive
- Limit request compression ratios
  - For Apache, use `mod_deflate`'s `DeflateInflateRatioLimit`, `DeflateInflateRatioBurst`, and `DeflateWindowSize` directives
The Search Continues

- Various protocols, i.e. SSH, FTP
- Fonts
- Videos
- Embedded devices
- Version control systems, i.e. Git, SVN

Anything that makes use of compression is a potential vector for this type of attack.
Tools

• GzipBloat
  https://github.com/cyberisltd/GzipBloat

• Burp Image Size Extension
  https://github.com/silentsignal/burp-image-size

• bomb.codes
  https://bomb.codes/
GzipBloat

Common Test Cases:

- 1TB 'HTML response'*, Gzip encoded
- 1TB 'HTML response'* with 4 rounds of Gzip encoding
- 1TB file download with 4 rounds of Gzip encoding
- 1TB file download, Gzip encoded
- 1G 'HTML response'*, Gzip encoded
- 1G file download, Gzip encoded
- 10G 'HTML response'*, Gzip encoded
- 10G file download, Gzip encoded
- 10G 'HTML response'* with 2 rounds of Gzip encoding
- 10G file download with 2 rounds of Gzip encoding
- 1Tb SDCH dictionary

* Obviously this isn't really HTML content - it will extract to a file full of zeros.
Image size matches request parameters

**Issue:** Image size matches request parameters  
**Severity:** Low  
**Confidence:** Firm  
**Host:** http://127.0.0.1:5000  
**Path:** /image.jpeg

**Note:** This issue was generated by the Burp extension: Image size issues.

**Issue detail**

The size of the image returned in the HTTP response (32 by 64) matches exactly the values of client-supplied parameters \(w\) and \(h\), respectively. This might mean that the server generates an image with dimensions specified by the client, which can lead to Denial of Service attacks if no limits are enforced.

**Remediation detail**

Limit the dimensions that can be requested as parameters of the request.

**Issue background**

While resizing images on the fly for generating thumbnails or previews might be useful, if the size is specified in parameters controlled by the client, an attacker can provide enormous numbers. While the attacker doesn't need to invest resources in such an attack, the server might allocate the required pixel buffer (resulting in out of memory situations) and/or perform calculations that scale with the size of the image (resulting in hogging the server CPU).
A decompression bomb is a file designed to crash or render useless the program or system reading it, i.e. a denial of service. The following files can be used to test whether an application is vulnerable to this type of attack.

When testing, it’s always better to start small and work your way up. Starting with the largest file available can seriously harm an application or system — so use these bombs with caution.

All files have been bzipped to work around Github’s 50MB file upload restriction. Groups of files have been zipped then bzipped. Remove these additional encodings prior to testing.

"When you see something that is technically sweet, you go ahead and do it and you argue about what to do about it only after you have had your technical success. That is the way it was with the atomic bomb."

— J. Robert Oppenheimer

### Archives

<table>
<thead>
<tr>
<th>Format</th>
<th>10GB</th>
<th>30GB</th>
<th>50GB</th>
<th>100GB</th>
<th>200GB</th>
<th>300GB</th>
</tr>
</thead>
<tbody>
<tr>
<td>7z</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bzip2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gzip</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LZ4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LZFSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Mitigation Summary

• **Restrict resources** – place limits on processes and their children
• **Don’t rely on size alone** – check image dimensions prior to rendering
• **Restrict file size output** – verify that the output file size won’t max out storage
• **Limit number of extracted files** – calculate the file total to ensure that storage/processing power won’t be overloaded
• **Perform dynamic testing** – always verify mitigations via manual testing to ensure that they are functioning properly

Archive bombs are decompression bombs, but not all decompression bombs are archive bombs.
Questions?
@bones_codes | cara.marie@nccgroup.com