爬虫搞崩网站后，程序员自制“Zip炸弹”反击，6刀服务器成功扛住4.6万请求

CSDN 2025年05月01日 08:51 江苏

在这个爬虫横行的时代，越来越多开发者深受其害：有人怒斥 OpenAI 的爬虫疯狂“偷”数据，7 人团队十年心血的网站一夜崩溃；也有人被爬虫逼到极限，最后只好封掉整个巴西的访问才勉强止血。但本文作者却走了一条完全不同的路——他靠一己之力，用一台每月仅需 6 美元的小破服务器，成功扛下了 Hacker News 热榜带来的流量海啸，还顺手反制了那些恶意爬虫，用“zip 炸弹”让它们原地爆炸。而他是怎么做到的，我们不妨来看看。

原文：https://idiallo.com/blog/surviving-the-hug-of-death

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

作者 | Ibrahim Diallo 编译 | 苏宓

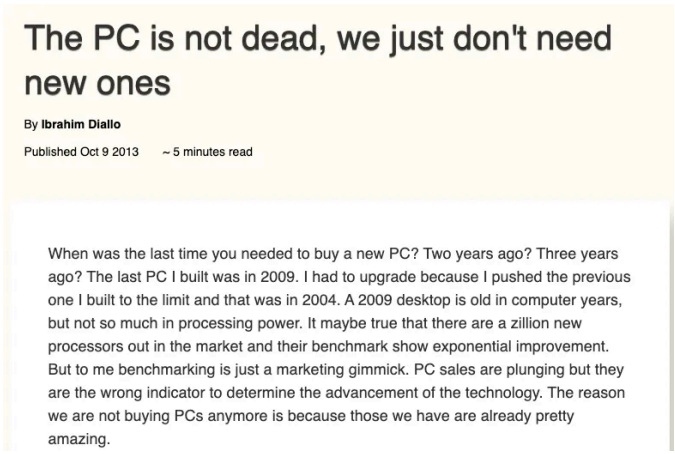
出品 | CSDN（ID：CSDNnews）

01

背景

我是 Ibrahim Diallo，一名住在加州的软件开发者。从 1992 年起，我就开始“折腾电脑”：破解系统、写代码，一转眼搞了几十年，到现在也还没停下。

平时我会写点博客，分享技术观察和一些个人思考。谁知多年前刚开始写没多久，就因为一篇意外“火出圈”的文章，遭遇了前所未有的流量暴击——直接冲上了 Hacker News 和 Reddit 热榜。



The PC is not dead, we just don't need new ones

By **Ibrahim Diallo**

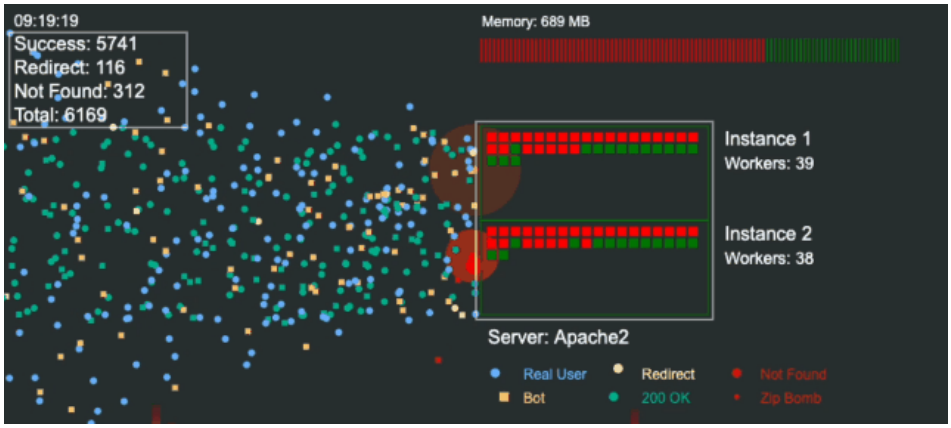
Published Oct 9 2013 ~ 5 minutes read

When was the last time you needed to buy a new PC? Two years ago? Three years ago? The last PC I built was in 2009. I had to upgrade because I pushed the previous one I built to the limit and that was in 2004. A 2009 desktop is old in computer years, but not so much in processing power. It maybe true that there are a zillion new processors out in the market and their benchmark show exponential improvement. But to me benchmarking is just a marketing gimmick. PC sales are plunging but they are the wrong indicator to determine the advancement of the technology. The reason we are not buying PCs anymore is because those we have are already pretty amazing.

文章登上 HN 首页的那一刻，我那台小服务器瞬间崩溃了。请求像潮水一样涌来，Apache 服务器吃力地运转，我坐在电脑前一次次重启它，就像拿个喷水枪去灭森林大火——完全招架不住。这种“被热度压垮”的场面，在互联网圈子里还有个形象的说法：“死亡之拥”（Hug of Death）。

到底访问量有多猛、服务器压力有多大？说实话，还真挺难用语言形容。

直到今年二月，我又写了一篇文章，它没多久后又登上了 Hacker News 第一名。不过这次我有备而来：提前保存了服务器的日志，还专门做了一段可视化动画，展示那台**每月只花 6 美元的小服务器**，是怎么扛住这波流量洪水的。



上面这个动画里，每一个网页请求都用一个小圆点表示，朝服务器移动。右下角还有图例说明：

- 机器人 vs 真实用户：通过 user-agent 判断。带有 “bot” 的一般是正规机器人，其他的就靠一些启发式规则来判断。
- 响应类型：

- 200 OK（是一种 HTTP 状态码，表示网页请求成功了，服务器已经正常返回了你要的内容）：请求成功

- 重定向：用晃动的小圆点表示

- 404 Not Found：红点会掉出屏幕

- zip Bombs：这个后面再解释

// 02

服务器配置

随着文章爆火，尽管现场一度混乱，我那台每月只花 6 美元的小服务器却始终坚挺。它的配置非常简陋：只有 1GB 内存，跑着 Apache 2 和一个简单的 MySQL 数据库。没有云服务、没有自动扩容、也没有负载均衡器，全靠轻量化配置和合理的缓存策略“硬扛”流量洪峰。

服务器配置如下：

- 主机服务商：DigitalOcean（1GB 内存）
- Web 服务器：Apache 2

- 系统环境：Ubuntu + PHP
- 数据库：MySQL
- 月费用：$6

我的博客是基于一个自建框架，大部分页面内容都提前缓存到了 memcached 中。数据库只在每小时查询一次页面，大大减轻了负担。这套方案在过去几次流量高峰时也都撑住了。

来看一下这次文章爆火的时间线：

- 下午 4:43（PST）— 文章提交到 Hacker News
- 下午 4:53（PST）— 登上首页，机器人蜂拥而至
- 下午 5:17（PST）— 成为 Hacker News 第一，洪水般的访问量涌入
- 晚上 8:00（PST）— 管理员改了标题（原因不明），流量断崖式下跌
- 凌晨 3:56（PST）— 一只机器人扫描了 300 个 URL，试图找漏洞
- 上午 9:00（PST）— 来自 Mastodon 网络的新一波流量暴增
- 上午 9:32（PST）— 遭遇大规模垃圾攻击：一分钟内接收约 4000 个请求，几乎全是广告
- 下午 4:00（PST）— 24 小时内，**服务器共处理 46,000 个请求**

关键点来了：服务器从头到尾都没宕机，甚至 CPU 使用率连 16% 都没到！

不过你在动画中可能注意到一个奇怪现象：明明是 1GB 内存的小服务器，怎么内存始终被占了一半左右？原因其实很简单——是 MySQL 在“背锅”。

当年博客刚上线时，我曾雄心勃勃地记录下每一条请求，把它们写入数据库日志表，用来追踪哪些文章最受欢迎。这招在当时确实挺实用，但 12 年过去，数据越堆越多，想从中查点东西反而变成了高成本操作。

这次流量冲击之后，我备份了日志表，然后彻底删掉它。也是时候跟那段历史告个别了。

对了，你可能在可视化动画里看到一些“小爆炸”效果，现在就来解释下那是怎么回事...

有一天，我偶然发现有个网站在实时偷我博客的内容：每次有人访问他们的页面，他们就立刻爬取我最新的文章，删掉我的名字和品牌标识，然后假装是他们自己写的。

一开始，我试着“手动反击”——故意喂他们一些假数据，让他们搬错内容。但没过多久，我就觉得这种方式太麻烦，干脆拿出了我的秘密武器：“zip 炸弹”。

这个“炸弹”的工作原理是这样的：当他们的爬虫访问我的网站时，我就返回一个看起来没什么问题的小压缩文件。他们的服务器会乖乖下载并尝试解压。结果呢？几 GB 的“垃圾”文件瞬间释放，系统直接崩了。

就这样，游戏结束。

03

什么是“zip 炸弹”？为什么能“反杀”爬虫？

这些年，“zip 炸弹”成了我对付各种恶意爬虫（比如偷内容、探测漏洞、发垃圾信息的自动程序）最有效的工具。

要知道，互联网上的流量，其实大部分都来自这些机器人。有的是“友好”的，比如搜索引擎、RSS 订阅器，或者现在很流行的大模型训练用爬虫。但也有很多是恶意的：像发送广告、偷文章，甚至入侵网站的脚本。我曾经在工作中就遇到过：有个爬虫发现我们 WordPress 的漏洞，在服务器里偷偷植入恶意代码，让网站变成黑客控制的攻击工具；还有一次，我的网站被爬虫刷满垃圾内容，结果直接被 Google 搜索下架。

从那以后我就意识到，必须对这些爬虫进行“反击”，zip 炸弹就是我找到的最佳方案之一。

简单说，zip 炸弹是一种超小的压缩文件，解压之后却会变成一个巨大的文件，能让处理它的系统崩溃。

在互联网早期，gzip 压缩是很早就被引入的一个功能。由于当时网速慢、信息密度高，人们希望尽可能压缩数据再传输。比如一个 50KB 的 HTML 文件，经过 gzip 压缩后可能只剩 10KB，能节省 40KB 的传输量。在拨号上网时代，这意味着页面加载时间从 12 秒减少到 3 秒。

这种压缩技术同样可以用于传输 CSS、JavaScript 甚至图像文件。Gzip 的优势在于它快速、简单，而且能显著提升浏览体验。当浏览器发出请求时，会在请求头中表明自己支持压缩。如果服务器也支持，就会返回经过压缩的数据版本。

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

而爬虫程序也会这么做，它们一样想节省资源。**我正是利用这一点“反制”它们。**

04

反击过程：用压缩包“让你解压到崩溃”

在我的博客上，经常会遇到一些机器人在扫描安全漏洞，通常我不会理会。但当我发现它们试图注入恶意攻击，或是在试探服务器响应时，我就会返回一个看起来正常的响应（比如 200 OK），并提供一个经过 gzip 压缩的文件。

我会提供一个从 1MB 到 10MB 不等的文件，它们也会欣然接受。通常一旦它们接收了这个文件，我就再也没见过它们出现。为什么？因为它们在解压文件后直接崩溃了。

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

具体发生的事情是这样的：机器人收到文件后，会读取文件头，发现这是一份压缩文件。于是它们尝试解压这个 1MB 的文件，以查找其中的内容。但文件开始不断扩展、扩展、再扩展，直到内存耗尽、服务器崩溃为止。这个 1MB 的压缩文件实际上会被解压为 1GB，大多数机器人在这一过程中就会崩溃。如果遇到那些死缠烂打的脚本，我就给它们发送 10MB 的版本，它会解压成 10GB，直接“秒杀”。

在告诉你如何制作 zip Bomb 之前，我得提醒一句：**这确实可能会导致你自己的设备崩溃甚至损毁，继续操作需自担风险。**

下面是创建 zip Bomb 的方法：

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

这个命令的作用如下：

- dd：用于拷贝或转换数据；
- if：指定输入文件，这里是 /dev/zero，一个会不断生成零字节数据的特殊设备；
- bs=1G：设置块大小为 1GB，意味着每次处理 1GB 数据；
- count=10：处理 10 个这样的块，总共生成 10GB 的零数据；
- 然后我们将这些数据通过管道传给 gzip，生成一个压缩文件 10GB.gz，压缩后大小大约是 10MB。

在我的服务器上，我写了一个中间件程序，用来自动识别哪些请求是恶意的。

我维护了一个黑名单 IP 列表，专门记录那些反复扫描整个网站的地址。同时还有一些启发式策略来检测垃圾信息发送者。很多垃圾脚本会在发完内容后再次访问页面，看内容是否成功插入，我正是利用这个行为模式来识别它们。

当系统判断出请求是恶意的，就触发如下逻辑：

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

就是这么简单。我只需要“送出”一个 10MB 的文件，就可能让对方服务器宕机。如果哪天我的文章突然大火、访问量激增，我就把炸弹换成 1MB 的“轻量版”，对付低端爬虫也完全足够。

// 05

Zip Bomb 并不是万能的

最后再补充一点：zip 炸弹并不是万能的。

高级一点的爬虫可以识别压缩文件，提前做限制，比如只读取一部分或禁止解压，这样就能绕过炸弹。不过那些“低级无脑乱爬”的脚本，就非常容易中招了。

而我需要的，就是一个成本低、效果好的防御手段。对于这种级别的威胁来说，“zip 炸弹”已经绰绰有余。

推荐阅读：

▶“一行代码让iPhone瞬间「变砖」！”发现iOS致命漏洞，开发者喜提12.7万元赏金

▶“少写一行代码，5分钟狂刷一次下载，开发者8000美元就这么烧没了！”

▶DeepSeek：放假是不可能放假的。671B 新模型已上线，开发者喜提“五一数学题”！


PROGRAMMER
新程序员
2024年度
中国开发者
调查报告
独家巨献
22年技术见证 《新程序员》编辑部 编著
.008
Daniel Jackson
Daniel Povey
Richard Vencu
Cay Horstmann
林咏华
张奇
潘欣
曾国洋
张俊林
李建忠
陈鑫
汪晟杰
天猪
刘政宁
张路宇
刘米东
吴岸城
Shawn Xiong
大模型驱动
软件开发
杨龙辉
姜伟
何文斯
黄佳
吴宗寰
陈晓涛
董卫强
直面“AI取代程序员”论，破解软件开发智能化的四大误区
世界级软件巨擘的亲身体会与思考
ACM Fellow、Kaldi之父、Stability AI MLOps负责人、Java经典著作作者齐聚
阿里巴巴、腾讯、字节跳动、京东、美团、蚂蚁、360、微博、清华大学、复旦大学、
智源人工智能研究院等25+组织及资深技术专家的最新实践经验
扫码立即订阅 ▶▶▶▶▶▶