

常见 IP 碎片攻击详解

January 11, 2002

Sinbad Technical Publications
Website: <http://sinbad.zhoubin.com>
© Copyright 2002, All Rights Reserved

本文简单介绍了 IP 分片原理，并结合 Snort 抓包结果详细分析常见 IP 碎片攻击的原理和特征，最后对阻止 IP 碎片攻击给出一些建议。希望对加深理解 IP 协议和一些 DoS 攻击手段有所帮助。

1. 为什么存在 IP 碎片

链路层具有最大传输单元 MTU 这个特性，它限制了数据帧的最大长度，不同的网络类型都有一个上限值。以太网的 MTU 是 1500，你可以用 `netstat -i` 命令查看这个值。如果 IP 层有数据包要传，而且数据包的长度超过了 MTU，那么 IP 层就要对数据包进行分片（fragmentation）操作，使每一片的长度都小于或等于 MTU。

我们假设要传输一个 UDP 数据包，以太网的 MTU 为 1500 字节，一般 IP 首部为 20 字节，UDP 首部为 8 字节，数据的净荷（payload）部分预留是 $1500-20-8=1472$ 字节。如果数据部分大于 1472 字节，就会出现分片现象。

IP 首部包含了分片和重组所需的信息：

```
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|           Identification   |R|DF|MF|           Fragment Offset           |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|<-----16----->|<---3--->|<-----13----->|
```

Identification: 发送端发送的 IP 数据包标识字段都是一个唯一值，该值在分片时被复制到每个片中。

R: 保留未用。

DF: **Don't Fragment**, “不分片”位，如果将这一比特置 1，IP 层将不对数据报进行分片。

MF: **More Fragment**, “更多的片”，除了最后一块外，其他每个组成数据报的片都要把该比特置 1。

Fragment Offset: 该片偏移原始数据包开始处的位置。偏移的字节数是该值乘以 8。

另外，当数据报被分片后，每个片的总长度值要改为该片的长度值。

每一 IP 分片都各自路由，到达目的主机后在 IP 层重组，请放心，首部中的数据能够正确完成分片的重组。你不禁要问，既然分片可以被重组，那么所谓的碎片攻击是如何产生的呢？

2. IP 碎片攻击

IP 首部有两个字节表示整个 IP 数据包的长度，所以 IP 数据包最长只能为 0xFFFF，就是 65535 字节。如果有意发送总长度超过 65535 的 IP 碎片，一些老的系统内核在处理的时候就会出现问題，导致崩溃或者拒绝服务。另外，如果分片之间偏移量经过精心构造，一些系统就无法处理，导致死机。

所以说，漏洞的起因是出在重组算法上。下面我们逐个分析一些著名的碎片攻击程序，来了解如何人为制造 IP 碎片来攻击系统。

3. ping o' death

ping o' death 是利用 ICMP 协议的一种碎片攻击。攻击者发送一个长度超过 65535 的 Echo Request 数据包，目标主机在重组分片的时候会造成事先分配的 65535 字节缓冲区溢出，系统通常会崩溃或挂起。ping 不就是发送 ICMP Echo Request 数据包的吗？让我们尝试攻击一下吧！不管 IP 和 ICMP 首部长度的了，数据长度反正是多多益善，就 65535 吧，发送一个包：

```
# ping -c 1 -s 65535 192.168.0.1  
Error: packet size 65535 is too large. Maximum is 65507
```

不走运，看来 Linux 自带的 ping 不允许我们做坏事。:(

65507 是它计算好的： $65535 - 20 - 8 = 65507$ 。Win2K 下的 ping 更抠门，数据只允许 65500 大小。所以你必须找另外的程序来发包，但是目前新版本的操作系统已经搞定这个缺陷了，所以你还是继续往下阅读本文吧。

顺便提一下，记得 99 年有“爱国主义黑客”（“红客”的前辈）发动全国网民在某一时刻开始 ping 某美国站点，试图 ping 死远程服务器。这其实是一种 ping flood 攻击，用大量的 Echo Request 包减慢主机的响应速度和阻塞目标网络，

原理和 ping o' death 是不一样的，这点要分清楚。

4. jolt2

`jolt2.c` 是在一个死循环中不停的发送一个 ICMP/UDP 的 IP 碎片，可以使 Windows 系统的机器死锁。我测试了没打 SP 的 Windows 2000，CPU 利用率会立即上升到 100%，鼠标无法移动。

我们用 Snort 分别抓取采用 ICMP 和 UDP 协议发送的数据包。

发送的 ICMP 包：

```
01/07-15:33:26.974096 192.168.0.9 -> 192.168.0.1
ICMP TTL:255 TOS:0x0 ID:1109 IpLen:20 DgmLen:29
Frag Offset: 0x1FFE Frag Size: 0x9
08 00 00 00 00 00 00 00 00 .....

```

发送的 UDP 包：

```
01/10-14:21:00.298282 192.168.0.9 -> 192.168.0.1
UDP TTL:255 TOS:0x0 ID:1109 IpLen:20 DgmLen:29
Frag Offset: 0x1FFE Frag Size: 0x9
04 D3 04 D2 00 09 00 00 61 .....a

```

从上面的结果可以看出：

- * 分片标志位 MF=0，说明是最后一个分片。
- * 偏移量为 0x1FFE，计算重组后的长度为 $(0x1FFE * 8) + 29 = 65549 > 65535$ ，溢出。
- * IP 包的 ID 为 1109，可以作为 IDS 检测的一个特征。
- * ICMP 包：
 - 类型为 8、代码为 0，是 Echo Request；
 - 校验和为 0x0000，程序没有计算校验，所以确切的说这个 ICMP 包是非法的。
- * UDP 包：
 - 目的端口由用户在命令参数中指定；
 - 源端口是目的端口和 1235 进行 OR 的结果；
 - 校验和为 0x0000，和 ICMP 的一样，没有计算，非法的 UDP。
 - 净荷部分只有一个字符'a'。

`jolt2.c` 应该可以伪造源 IP 地址，但是源程序中并没有把用户试图伪装的 IP 地址赋值给 `src_addr`，不知道作者是不是故意的。

7. 参考资料

- [1] TCP/IP Illustrated Volume 1 : The Protocols
- [2] Microsoft Security Bulletin MS00-029:
<http://www.microsoft.com/technet/security/bulletin/ms00-029.asp>
- [3] BugTraq Mailing List, "Analysis of jolt2.c(MS00-029)":
<http://www.securityfocus.com/archive/1/62011>
- [4] <http://www.attrition.org/security/denial/w/teardrop.dos.html>
- [5] <http://packetstormsecurity.org/0005-exploits/jolt2.c>
- [6] <http://packetstormsecurity.org/Exploit Code Archive/teardrop.c>