第 32 卷 第 21 期 计 算 机 工 程 2006 年 11 月
Vol.32 No.21 Computer Engineering November 2006
·安全技术· 文章编号：1000—3428(2006)21—0164—03 文献标识码：A 中图分类号：TP393.08

IP 追踪中 PPM 算法的改进研究

陈星星，徐红云
(湖南大学计算机与通信学院，长沙 410082)

摘 要：概率包标记(PPM)是对拒绝服务(DoS)攻击进行 IP 追踪的一种实用而有效的方法。文章提出通过利用 TTL 域对原有 PPM 方案进行改进，减少了路径重构所需的数据包数量，提高了路径重构的效率。
关键词：IP 追踪；分布式拒绝服务；概率包标记；TTL

Research on Improvement of PPM for IP Traceback

CHEN Xingxing, XU Hongyun
(College of Computer and Communications, Hunan University, Changsha 410082)

【Abstract】 Probabilistic packet marking(PPM)is a practical and effective method for IP traceback of denial-of-service(DoS)attack. The probabilistic packet marking scheme is improved by using TTL, and the new scheme can decrease the number of packet the path reconstruction needs, and can increase the efficiency of path reconstruction.
【Key words】 IP traceback; Distributed denial of service(DDoS); Probabilistic packet marking; TTL

拒绝服务攻击(DoS)通过消耗计算机网络资源使得 Internet 站点不能正常的提供或者接受服务。分布式拒绝服务攻击(DDoS)是指大量的攻击主机同时对受害者发起拒绝服务攻击，其攻击强度比单攻击者情形要大得多，而且由于攻击者的分布式特性，使得预防和消除这类攻击更加困难。DDoS 攻击利用 IP 协议设计上的缺陷，通常采用伪造的 IP 源地址，这样就使得确定 IP 源十分困难。如何找出真正的攻击者，即 IP 追踪(IP Traceback)问题，成为当前 Internet 安全领域一个比较活跃的课题。

在防范 DDoS 攻击的工作中，找到真正的攻击者，并且将其彻底消除是非常困难的。但是攻击者需要大量的傀儡机，找到这些傀儡机也是很有意义的。受害者可以对傀儡机发来的数据流采取过滤、限流等措施从而减少受害的程度，同时受害者也可以通知傀儡机的管理员，从而堵塞傀儡机的漏洞、加强傀儡机的安全(这对攻击者而言也是个损失，因其可能因此而失去对傀儡机的控制，从而失去利用傀儡机资源的能力)，甚至从傀儡机进一步追踪到真正的攻击者。

最早Savage[1]等研究的概率包标记方案就是为了追踪攻击者或傀儡机而提出的，后来有许多研究者提出一些改进。本文是在他们的研究基础上提出的，以充分利用IP数据包格式中使用频率较低的标识域，并且减少在攻击路径重构过程中所需的数据包数及边界碎片组合的次数，从而使受害者能及早地追踪到攻击者(傀儡机)并及时对DoS攻击予以响应。

1 相关研究工作

概率包标记的基本思想是路由器以某概率将经过的数据包进行标记，记录该路由器的部分 IP 信息，受害者收到足够的数据包后，根据包中的碎片信息重构出数据包经过的路径。在 DoS 攻击中，傀儡机不停地向受害者请求服务，因此一般情况下，受害者都能收到足够的数据包。

由于数据包在途中经分段(fragment)处理的情况是很少出现的(不超过 0.25%)，因此 IP 头中的识别号域(Identification field)也很少使用。于是，Savage 等人建议将路径信息写入到 16bit 的识别号域中。在文献[1]中，路由器的 IP 地址及另外的 32bit 校验码共 64bit 被分成 8 块，每块 8bit，以 0－7 对其编号(称为偏移值)。为了便利进行路径的重构，还需要一个距离域表示路由器到受害者之间的距离，由于路径极少有超过 25 跳(hop)的，因此 5bit 的空间就够了。当一个路由器标记一个数据包时，其随机地从其 8 个分块中选取 1 块(8bit)，连同对应的偏移(3bit)，以及距离(5bit，在标记时置 0)填入该数据包的标记域(即识别号域)中。当一个路由器选择不标记一个数据包时，它先检查距离域的值是否为 0，如是，则其把自己的与标记域中偏移量对应的分块与包中已有的分块异或再填入原有位置(此时记录的是边界 id 的碎片)，然后把距离增 1；如果距离不是 0，则它只需把距离增 1。这样一来，数据包中的标记信息实际上是 2 个相邻路由器之间的边(或连接)的信息。在这个方案中，至少需要 8 个数据包才能传送一个路由器(或边)的完整信息。

路径重构时，受害者选取 8 个距离值相同而偏移值不同的碎片进行组合，并使用由边界 id 计算出的 hash 值与碎片所携带的 hash 值比较，相同则为有效边界，反之则丢弃此边界。当距离值为 0 时，计算出的边界 id 就是 IP，所以通过与上游邻近边界相异或，逐步得出上游路由器的 IP，从而重构出完整的路径。这种包标记的思想出现得最早，而且许多研究者在此基础上进行研究，因此被称为基本包标记(Basic PPM，BPPM)。

高级包标记和带认证的包标记[2]与此类似，主要的不同在于其 8 个分块不再是IP地址和校验码，而是IP地址的 8 个不同的hash值(共有 8 个不同的hash函数)。这 2 种方法在重构

基金项目：湖南大学科学基金资助重点项目(521101626)
作者简介：陈星星(1981－)，男，硕士生，主研方向：网络信息安全；徐红云，副教授
收稿日期：2005-12-27 **E-mail：**chenxx0@126.com

攻击路径时，采用网络拓扑信息来验证得到的IP，与基本包标记方法相比，数据包中不需要记录边界id的hash值，因此可以节省空间来记录别的信息；而网络拓扑信息则很容易获得。此外，一些研究人员在基本包标记和高级包标记的基础上，提出了动态(或自适应)概率包标记方案，以降低受害者在重构攻击路径时对数据包数目的要求[4]，如图 1 所示。

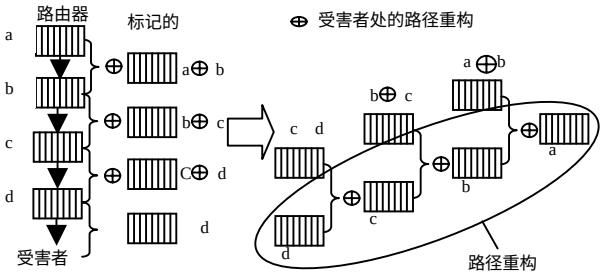

图 1 基本包标记的包标记和路径重构示意图

在文献[4]中，使用了 TTL 域和 Fragment Offset 域。在该算法中，当某路由器接收到一数据包时，读取包中的 TTL 值，根据它来确定标记该包的概率，确定标记时，在其 TTL 域写入 64。使用 Fragment Offset 域会带来很大的负面影响，在终端计算机收到数据包时，会判断该域的值是否为 0，如果不为 0，则认为该包为数据包碎片，并等待其他数据包碎片，这样很容易导致系统崩溃。使用 TTL 的负面影响相对较小，只会使 traceroute 等应用失效。该方案称为自适应概率包标记(Adaptive PPM，APPM)。

在基本包标记方法中，重构路径时，需要 8 个有效碎片才能组合出一个边界id。由于包标记的概率一般都很小，而且已经标记过的包可能被下游路由器的标记覆盖，因此受害者需要大量的数据包。当有多个分布式的攻击者时，offset 为 0 的fragment有多个，设为k_0个，offset为 1,2,…的也有多个，这时会有 $\prod_{i=0}^{7} k_i$ 次组合，这些高计算量会给受害者带来很高的负载。

2 概率包标记方案的改进

在基本包标记、高级包标记以及一些改进方案中，都有一个 distance 域，用来表示标记数据包的路由器与受害者之间的距离。由于 TTL 域的值每经过一个路由器就会减少 1，这样就可以记录数据包经过的路由器数目，因此可以利用 TTL 域代替 distance 域，使得可以节省更多的空间存储路由器的信息。

路径重构时，采用网络拓扑信息来验证IP的方法[2]。跟高级包标记类似，也假设有一张上游路由器的分布图，称为*G*m。*G*m为有向无环图(Directed Acyclic Graph，DAG)，受害者为*G*m的根节点。

2.1 IP 数据包头编码方案

由于IP包头中flags域的第 1 位为保留位，在这里使用了这 1bit[3]。加上Identification域TTL域，整个利用的IP数据包头中的长度为 25bit。具体编码格式如图 2 所示。

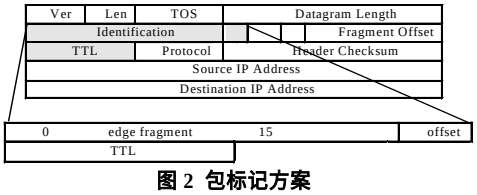

图 2 包标记方案

它们的分布和作用如下：

Edge fragment：16bit，用于记录边界 id 的碎片信息。这样可以使得 2 个数据包就可以记录 1 条边界。

Offset：1bit，用 0、1 标识边界 id 的 2 段 fragment。

2.2 算法实现

包标记算法和路径重构算法如下：

```
Marking procedure at router R:
for each packet w
  let x be a random number from [0..1)
  let o be a random integer from [0,1]
  let f be the fragment of R at offset o
  if x < p then
          write f into w.frag
          write 64 into w.TTL
          write o into w.offset
else
          if w.TTL = 64 then
            let f be the fragments of R at offset w.offset
            write f⊕w.frag into w.frag
Path reconstruction procedure at victim v:
let FragTbl be a table of tuples (frag, offset, distance)
let G be a tree with root v
let edges in G be tuples (start, end, distance)
let maxd := 0
let last := v
for each packet w from attacker
   FragTbl.Insert( w.frag, w.offset, 65 – w.TTL)
   if (65 – w.TTL) > maxd then
      maxd := 65 – w.TTL
for d := 0 to maxd
   for all ordered combinations of fragments with different offset at distance d
      construct edge z
      if d ≠ 0 then
        z := z⊕last
        for each child u of last in Gm
          if u = z then
            insert edge (u, last, d) into G
            last := u;
remove any edge (x, y, d) with d ≠ distance from x to v in G
extract path (Ri..Rj) by enumerating acyclic paths in G
```

在包标记算法中，将边界 id 信息分成两部分，由 2 个数据包来携带。当路由器转发某个数据包时，以概率 *p* 标记该数据包。如果该路由器确定标记该数据包，随机选取一个 offset 值，将 offset 值和对应的 fragment 写入数据包，并在 TTL 域写入 64(相当于在 distance 域写入 0)；否则，当检测到该数据包的 TTL 域的值为 64 时(相当于检测 distance 域是否为 0)，说明该包刚被上一路由器标记，选取 offset 相同的 fragment，与包中的 fragment 进行异或，再覆盖写入数据包中。被标记过的数据包 edge 域中的值为相邻路由器的 fragment 的异或值，除非标记该包的路由器与受害者的距离为 1hop。由于在现有网络中，数据包每经过一个路由器，其 TTL 值就会减 1，因此在算法中无需实现这一步。

在路径重构时，选取 2 个 TTL 值相同而 offset 值各不相同的 fragment 进行组合，然后通过网络拓扑信息来验证得到的 IP，若得到的 IP 合法，则记录下来，否则进行下一个碎片组合。由于 $a⊕b⊕a=b$，可以从离受害者距离为 1hop(即 65 –

w.TTL 为 1)的路由器开始，逐步将前边的路由器计算出来，直到第 1 个路由器。

2.3 理论分析

基本包标记算法在路径重构中，至少需要 8 个有效数据包才能组合出 1 个边界id，而本文因为利用了网络拓扑信息来验证IP的有效性，所以只要 2 个有效数据包就能组合出 1 个边界id。设需要k个fragment才能组合出 1 个边界id，路由器以概率p标记数据包，当重构出长度为d的路径时，需要的数据包数量的期望值为[2]

$$E(X) < \frac{k \times \ln(kd)}{p(1-p)^{d-1}} \tag{1}$$

例如，在标记概率为 4%的情况下，重构 1 条长度为 10hop 的攻击路径，在基本包标记中 k=8，至多需要 1 300 个数据包；而在改进方案中 k=2，只需要 220 个数据包，就能重构出这条路径。

在基本包标记中 k 为 8， 改进后的方案中 k 为 2。用 EPPM(即 Enhanced-PPM)表示改进后的方案，因此

$$\frac{E(X)_{EPPM}}{E(X)_{BPPM}} < \frac{1}{4} \tag{2}$$

可以看出，改进方案在重构相同长度路径时，所需数据包数目不到基本包标记方案所需的 1/4。

路径重构时，会有大量的碎片组合运算。设k_i表示距离值d、offset值为i的fragment数目，基本包标记方案中碎片组合次数为 $\prod_{i=0}^{7} k_i$ ；本方案中，碎片组合次数为k_0k_1。这样大大减少了碎片组合次数，因此能减少路径重构时的计算量。

3 实验及分析

为了测试改进方案的性能进行了模拟试验。实验的数据集来自Lucent Bell实验室[5]。这个数据集包含从一个源节点到 103 402 个目标的 709 310 条不同路径，这些路径基本上可以覆盖整个Internet。在这些实验中，将这个源节点作为受害者，数据集作为受害者上游网络拓扑结构信息。随机选择一些目标节点作为攻击者发动攻击，然后模拟路由器的功能来标记攻击数据包，最后模拟受害者通过那些标记来构造出攻击图。为测试构建攻击路径所需包的数目，在Gm中分别选取某一攻击路径，使其路径长度d的取值分别为d=1,2,…,30。对d的每一取值，发送包数目分别取为r=100,150,200,…，直到受害者正确构建此攻击路径为止，对每一d值重复做 50 次实验，取p=4%，所得结果如图 3 所示。

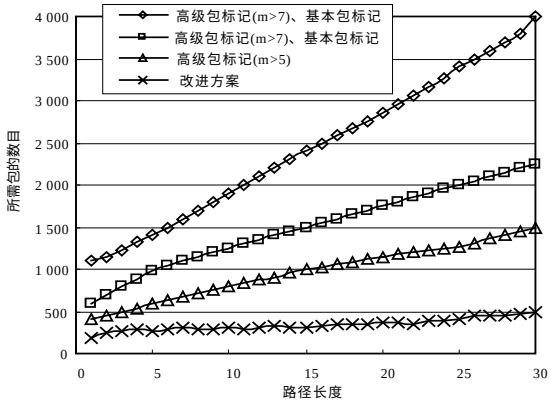

图 3 各种方案所需数据包数目比较

图 3 中的 m 表示高级包标记中确定一条边界 id 所需不同 hash 值的数据包数目。可以看出，改进方案和以前的方案相比，重构路径时要求的数据包数量要少得多，故在遭受攻击时只需要其他方案的少部分数据包，就能重构出攻击路径。

4 结论

本文通过使用 TTL 域对已有的包标记方案进行改进。理论分析和模拟实验结果表明，改进后的方案可以使受害者使用更少的数据包(少于基本包标记的 25%)就能重构出攻击路径。这样为受害者及早地对 DoS 攻击作出响应、减少攻击带来的危害创造了条件。

参考文献

1 Savage S, Wetherall D, Karlin A, et al. Practical Network Support for IP Traceback[C] Proceedings of the 2000 ACM SIGCOMM Conference, Stockholm, Sweden, 2000-08: 295-306.

2 Song D X, Perrig A. Advanced and Authenticated Marking Schemes for IP Traceback[C]. Proceedings of IEEE INFOCOM'01, Anchorage, Alaska, 2001-04: 878-886.

3 Belenky A, Ansari N. IP Traceback with Deterministic Packet Marking[J]. IEEE Communications Letters, 2003, 7(4).

4 Liang Feng, Zhao Xinjian, Yau D. Real Time IP Traceback with Adaptive Probabilistic Packet Marking[J]. Journal of Software, 2003, 14(5): 1005-1010.

5 Internet Mapping[Z]. http://cm.bell-labs.com/who/ches/map/dbs/index.html, 1999.

（上接第 163 页）

root 用户的权限，降低潜在的风险；另一方面，新模型可以赋予一部分特定的普通用户一些必要的特权，同时取消 root 用户程序的 setuid 位，从根本上解决 setuid 机制的安全问题。

另外，在 Linux 中，某些特权的组合足以对系统构成威胁，而将这些特权分割开来则又是安全的。在改进前的模型下，不得不通过去除某些特权来保证安全，这等于放弃了那些需要这些特权的服务。而在改进后的模型下，可以分割那些危险的特权组合，在保证系统安全的情况下继续使用相应的服务。

4 结论

root 用户拥有所有的特权是 Linux 的一个安全隐患。POSIX 的 Capabilities 模型试图通过分割 root 用户的特权来解决这个问题。然而，由于在 Linux 下可执行文件的权能模型并没有实现，问题并没有得到很好的解决。本文引入基于用户的 Capabilities 模型并给出 Linux 下的实现方案，真正实现了用户特权的分割，增强了 Linux 的系统安全。

参考文献

1 Andrew. Linux Capabilities FAQ 0.2[Z]. http://www.kernel.org.

2 Linux version 2.4.18（内核源代码）[Z]. http://www.kernel.org.

3 Bacarella M. Taking Advantage of Linux Capabilities[Z]. http: // www.linuxjournal.com.

4 李善平, 刘文峰, 李程远等. Linux 内核 2.4 版源代码分析大全[M]. 北京: 机械工业出版社, 2002.

5 巫晓明, 郭玉东. Linux 下的Capabilities安全机制的分析与完善[J]. 计算机应用, 2004, 24(6): 203-205.