ISSN 1000-0054 清华大学学报（自然科学版） 2014 年 第 54 卷 第 11 期 20/20
CN 11-2223/N J Tsinghua Univ (Sci & Technol), 2014, Vol. 54, No. 11 1502-1510

可增量部署、基于采样流的 IP 溯源方法

田红成[1,2]， 毕 军[1]， 王 虹[2]

（1. 清华大学 信息网络工程研究中心，北京 100084；2. 解放军第 309 医院 信息科，北京 100091）

摘 要：IP 溯源能用来查找攻击流量的源头和路径，但迄今为止，还没有因特网规模的溯源方法被实际部署。该文提出了一种基于采样流的部署点自治域（AS）级日志记录类溯源方法 SampleTrace。该方法利用路由器已存在的 xFlow 功能（sFlow、NetFlow 和 IPFIX）和边界网关协议（BGP）信息来实现溯源，并在因特网上建立覆盖网络以支持增量部署。该文从理论上分析了攻击流被成功溯源的概率与独立采样概率、攻击流所含攻击包数、被成功溯源的跳数 3 个因素的定量关系。根据 Bernoulli 大数定律，当使用 SampleTrace 对因特网中大量的攻击流进行溯源时，成功溯源的经验概率将接近于成功溯源的概率。SampleTrace 较好地解决了日志记录类溯源方法的增量部署难题。

关键词：计算机网络；IP 溯源；攻击流；日志；覆盖网络

中图分类号：TP 393 **文献标志码**：A

文章编号：1000-0054(2014)11-1502-09

Incrementally deployable IP traceback scheme based on sampled flows

TIAN Hongcheng[1,2], BI Jun[1], WANG Hong[2]

(1. Network Research Center, Tsinghua University, Beijing 100084, China;
2. Department of Information, The 309th Hospital of PLA, Beijing 100091, China)

Abstract: IP traceback can identify the attacking sources and attacking paths of malicious traffic, but Internet-scale IP traceback methods have not yet been deployed. This paper presents an incrementally deployable IP traceback scheme based on sampled flows (SampleTrace). SampleTrace uses existing xFlow (sFlow, NetFlow and IPFIX) functions and border gateway protocol (BGP) information to implement the traceback and builds an autonomous system (AS)-level overlay network for incremental deployment. Theoretical analyses show that the probability that a flow is successfully traced back through various AS-level hops quantitatively depends on the independently sampling probability and the number of packets in the flow. According to Bernoulli's law of large numbers, when a large number of attacking flows are practically traced back by SampleTrace, the successfully-traced back relative frequency will approach the successfully-traced back probability. SampleTrace solves the incremental deployment difficulty of logging-based traceback schemes.

Key words: computer network; IP traceback; attacking flow; logging; overlay network

因特网地址溯源的目标为确定攻击流量的源和攻击路径，但是大多数溯源方法难以部署在因特网中，因为需要在路由器上部署专门的软件或硬件，并且大多数方法难以增量部署。

为此，本文提出了可增量部署、基于采样流的溯源方法 SampleTrace。SampleTrace 使用边界网关协议（border gateway protocol, BGP）路由器目前具有的 xFlow 功能（sFlow[1]、NetFlow[2-3] 和 IPFIX[4-8]）和 BGP 信息来实现溯源，而不用在路由器上部署任何溯源软件和硬件。SampleTrace 在部署点自治域（autonomous system, AS）间通过上游逻辑邻居发现机制建立 AS 级的覆盖网络以支持增量部署。另外，本文为收集器（collector）设计了两种机制：时间同步机制和聚合机制。时间同步机制用于统一处理从不同的 BGP 路由器发送过来的采样信息，聚合机制用于统一处理由 sFlow 和 NetFlow(IPFIX)产生的不同粒度的采样信息。

SampleTrace 有 3 种部署激励：

1）部署点 AS 能够将溯源功能作为一项收费服务提供给其他 AS、终端用户或者入侵检测系统（instrusion detection system, IDS）；

2）对于一个末端部署点 AS（stub deployed AS），当该 AS 的用户被攻击时，SampleTrace 能确定攻击流从哪个（些）路由器的哪个（些）接口进入该 AS；然后，该 AS 的管理员可以在相关路由器的相关接口上采取过滤或者限流措施，来保护该 AS 的

收稿日期：2014-07-10
基金项目：高等学校博士学科点专项科研基金（20090002110026）；
国家“十一五”科技支撑计划（2008BAH37B02）
作者简介：田红成（1976—），男（汉），湖北，博士研究生。
通信作者：毕军，教授，E-mail: junbi@tsinghua.edu.cn

用户；

3）假如一个穿越 AS 能提供更多的服务，例如溯源服务，则该穿越 AS 对于潜在的客户 AS 将更有吸引力。

1 可增量部署基于流的溯源方法

SampleTrace 由 3 种机制组成：覆盖网络建立机制，攻击流的采样和记录机制，以及在覆盖网络中的溯源机制。

1.1 定义和假设

假设任意两个部署点 AS：AS_i 和 AS_j，如果存在一条从 AS_i 到 AS_j 的路由不穿越其他部署点 AS，则 AS_j 被称作 AS_i 的下游逻辑邻居；同样，AS_i 被称作 AS_j 的上游逻辑邻居。上游逻辑邻居和下游逻辑邻居统称为逻辑邻居，均为部署点 AS。

假定有任意两个 AS，AS_m 和 AS_p，如果 AS_m 通过一条物理链路连接到 AS_p，则 AS_m 被称为 AS_p 的物理邻居，AS_p 也被称为 AS_m 的物理邻居。另外，如果多个 AS 通过一个交换机互联，这些 AS 彼此之间称对方为物理邻居。

本文基于如下两个假设：

1）每个部署点 AS 在某一个组织的网站上注册和公开了它的 AS 号和溯源服务器（traceback server，TS）的 IP 地址。注册激励为：如果公开了它的 AS 号和溯源服务器的 IP 地址，则该 AS 的溯源服务可以作为收费服务提供给其他 AS、终端用户或者 IDS。因此，每个部署点 AS 的溯源服务器知道所有部署点 AS 的 AS 号和相应的溯源服务器的 IP 地址。

2）每个溯源服务器知道自己所属的 AS、该 AS 的所有 BGP 路由器、每个 BGP 路由器的所有外连接口、每个外连接口的对等接口的介质访问控制（media access control，MAC）地址，以及这些对等接口各自所属的 AS。

1.2 域内结构

图 1 是部署点 AS 的域内结构示意图。该 AS 的 BGP 路由器的外连接口开通了 xFlow 功能以对进入该 AS 的流量进行采样。BGP 路由器将采样的流量信息发送到收集器，收集器统一处理采样信息。如果一个部署点 AS 的规模和流量比较大，则在该 AS 中可以部署多个收集器。

每个部署点 AS 在功能上有一个溯源服务器，溯源服务器运行 BGP 协议，从它所属的 AS 的内部边界网关协议（internal border gateway protocol，iBGP）对等体学习 iBGP 路由，并根据 BGP 协议计算最佳路由。但是，该溯源服务器不宣告任何路由前缀，不转发任何流量。溯源服务器将它计算得到的所有最佳路由记录进历史路由信息表（historical route information base，HRIB）。

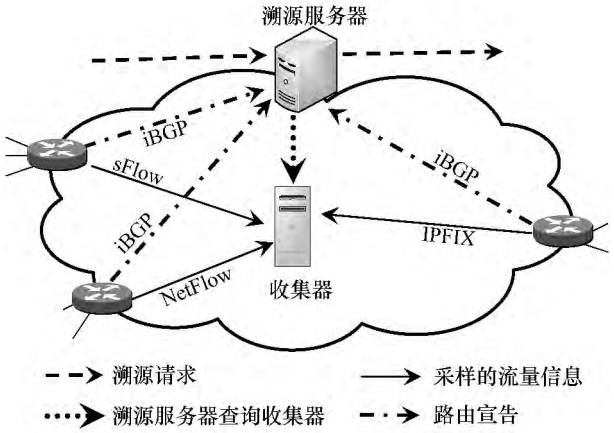

图 1 部署点 AS 内部结构示意图

溯源服务器向其他溯源服务器、终端用户或者 IDS 提供服务接口。当一个溯源请求发送给一个溯源服务器时，溯源服务器验证请求发送者的身份以及对溯源请求进行完整性检查，查询溯源服务器所属 AS 的收集器，向其他某些溯源服务器发送溯源请求。当一个溯源过程被一个溯源服务器终止之后，该溯源服务器将溯源结果发送给终端用户或 IDS。

1.3 建立 AS 级的覆盖网络

通过上游逻辑邻居发现，每个溯源服务器知道它所属的 AS 的上游逻辑邻居，这样便建立了 AS 级覆盖网络。依靠该 AS 级覆盖网络，SampleTrace 可以逐跳向上游逻辑邻居溯源一个攻击流。在下面的陈述中，假定 AS_i 是任意一个部署点 AS，AS_j 是 AS_i 的任意一个下游逻辑邻居。本文将以 AS_i 和 AS_j（溯源服务器分别为 TS_i 和 TS_j，TS_i 和 TS_j 的历史路由信息表分别为 $HRIB_i$ 和 $HRIB_j$）为例，来描述如何建立 AS 级的覆盖网络。

1.3.1 溯源服务器的预处理过程

AS_PATH 暗示了部署点 AS 之间的上、下游逻辑邻居关系。溯源服务器 TS_i 通过扫描 $HRIB_i$ 的 AS_PATH 维护着两种集合：AS_i 的下游逻辑邻居集和出现在 $HRIB_i$ 的 AS_PATH 中所有部署点 AS 的上游逻辑邻居集。根据第 1.1 小节的假设 1），

TS_i知道所有的部署点 AS 的 AS 号。AS_PATH 由一个或多个路径段组成,每个路径段是 AS_SET 或者 AS_SEQUENCE。一般来讲,AS_PATH 有 3 种典型的形式[9]:整个 AS_PATH 是 AS_SET,整个 AS_PATH 是 AS_SEQUENCE,AS_PATH 由两个路径段组成(前一个路径段为 AS_SEQUENCE,后一个路径段为 AS_SET)。对于这 3 种典型的 AS_PATH 形式,TS_i将进行不同的处理:

1) 如果整个 AS_PATH 是 AS_SET,在此 AS_PATH 中,所有的部署点 AS 都为 AS_i的下游逻辑邻居,出现在 AS_PATH 中的每个部署点 AS 把该 AS_PATH 中的其他部署点 AS 以及 AS_i作为它的上游逻辑邻居。

2) 如果整个 AS_PATH 是 AS_SEQUENCE,TS_i将从左到右查找 AS_PATH,假如第一个部署点 AS 被找到,则该 AS 为 AS_i的下游逻辑邻居。同样地,AS_i为此 AS 的上游逻辑邻居。然后,该查找过程继续。假如第 2 个部署点 AS 被找到,则第 1 个部署点 AS 为第 2 个部署点 AS 的上游逻辑邻居,第 2 个部署点 AS 为第 1 个部署点 AS 的下游逻辑邻居。查找过程一直进行下去,直到该 AS_PATH 的末尾。

3) 假如 AS_PATH 由两种路径段组成(前一个路径段为 AS_SEQUENCE,后一个路径段为 AS_SET),则查找过程类似于上面的两种情况。

图 2 是查找过程的示例。TS_i从 $HRIB_i$的每个 AS_PATH 中提取上、下游逻辑邻居关系,于是每个 TS_i知道 AS_i的下游逻辑邻居集 $S_d(AS_i)$和出现在 $HRIB_i$的 AS_PATH 中的所有其他部署点 AS 的上游逻辑邻居集。每个溯源服务器的预处理过程是相同的。

AS_1 非部署点 AS

AS_PATH

AS_1 AS_2 AS_3 AS_4

AS_5 AS_6 AS_7 AS_8 AS_9

AS_{10} AS_{11} AS_{12} | AS_{13} AS_{14}

AS_1 部署点 AS

种类

AS_SEQUENCE

AS_SET

前部分是 AS_SEQUENCE

后部分是 AS_SET

部署点 AS	AS_2	AS_4	AS_5	AS_9	AS_{11}	AS_{13}	AS_{14}
部署点 AS 的上游逻辑邻居	AS_i	AS_2	AS_i AS_9	AS_i AS_5	AS_i	AS_{11} AS_{14}	AS_{11} AS_{13}

AS_i 的下游逻辑邻居:AS_2, AS_5, AS_9, AS_{11}

图 2 TS_i推导上、下游逻辑邻居的例子

1.3.2 上游逻辑邻居发现机制

TS_i的预处理过程完成之后,TS_i维护着 AS_i的下游逻辑邻居集 $S_d(AS_i)$。

TS_i给 $S_d(AS_i)$每个成员 AS 的溯源服务器发送一个查询请求,询问哪些部署点 AS 是 AS_i的上游逻辑邻居。假定 AS_j是 AS_i的任意一个下游逻辑邻居,AS_j的 TS_j将从 TS_i接收到一个查询请求,由于 TS_j的预处理过程完成之后,TS_j维护着从 AS_j的角度来看 AS_i的上游逻辑邻居的集合 $S_u(AS_i, AS_j)$。TS_j把 $S_u(AS_i, AS_j)$发送给 TS_i。图 3 描述了 TS_i与 TS_j的通信过程。当 TS_i接收到 $S_d(AS_i)$中所有成员 AS 的溯源服务器的响应之后,TS_i将得到 AS_i的上游逻辑邻居的集合,则有

$$S_u(AS_i) = \bigcup S_u(AS_i, AS_j),\quad AS_j \in S_d(AS_i). \tag{1}$$

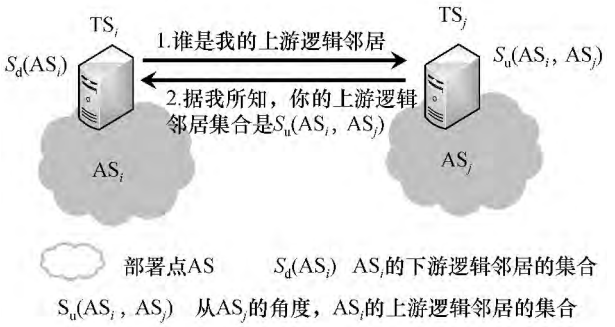

图 3 发现 AS_i 的上游逻辑邻居的交互过程

1.3.3 采样和记录攻击流

因为 IPFIX 类似于 NetFlow,下面本文将只讨论 sFlow 和 NetFlow。在收集器统一处理采样信息之前,两个问题必须解决:1) xFlow 输出包的时间戳基于不同 BGP 路由器的时钟,这些时钟是不容易被同步的;2) sFlow 和 NetFlow 输出包在粒度上是不同的。在 SampleTrace 中,本文分别设计了时间同步机制和聚合机制来解决上面的两个问题。

本小节要实现的目标为:从部署点 AS 的收集器的角度,攻击流什么时候和从哪里进入该部署点 AS 将被统一以溯源记录的形式存储在收集器端。

1) 时间同步机制。

xFlow 输出包被不同的 BGP 路由器打上了时间戳,但不同的 BGP 路由器的时钟很难被同步。在 SampleTrace 中,xFlow 输出包的时间戳被收集器统一修改,从而与收集器的时钟同步。这样,从不同 BGP 路由器发送到收集器的流量信息将按照时间

序存储在收集器端。

在sFlow输出包头中，有一个时间戳uptime，表明BGP路由器发送该输出包给收集器的时刻。在一个NetFlow输出包中（包括一个NetFlow输出包头和一个或多个流记录），有3种主要的时间戳：第1个时间戳sysUpTime，位于NetFlow输出包头中，表明BGP路由器发送该输出包给收集器的时刻；第2时间戳first switched和第3个时间戳last switched位于NetFlow的流记录中，表明一个流的第一数据包和最后一个数据包是什么时候在该路由器的接口上被观测到的。上面3个NetFlow输出包的时间戳都是基于BGP路由器的时钟。第2个和第3个时间戳相对于第1个时间戳的相对时间可以预先被计算出来。

在SampleTrace中，当收集器接收到一个sFlow输出包时，用收集器接收该输出包的时刻替换数据包头中的uptime。当收集器接收到一个NetFlow输出包时，用收集器接收该输出包的时刻替换数据包头中的sysUpTime。并且，对于NetFlow输出包，流记录中的first switched和last switched可以根据替换后的第1个时间戳值和预先计算出的相对时间相应地修改为收集器的时刻。

2）对sFlow包的聚合机制。

被xFlow采样和发送到收集器的流量信息，在粒度上是不同的。sFlow信息是包级别的，但是NetFlow信息是流级别的。具体来说，在采样数据包后，Netflow会将采样的数据包的包头和相关的路径信息聚合成流级别的信息，然后把它们发送给收集器。然而，sFlow在采样数据包后，只会作很少的处理，直接将采样的数据包头和相关的路径信息发送给收集器。

在SampleTrace中，流被定义为七元组：路由器的入接口索引、以太网帧的源MAC地址、源IP地址、目的IP地址、源端口、目的端口和协议。在接收到NetFlow输出包后，收集器将直接获得流级别的信息。在接收到一系列的sFlow输出包后，收集器将模拟路由器端的NetFlow处理过程，将采样的数据包头信息和相关的路径信息聚合成流级别的信息。

收集器从xFlow数据包头中可以提取发送路由器的IP地址。在SampleTrace中，存储在收集器端的每条溯源记录将包含下面的10个属性：发送路由器的IP地址、发送路由器的入接口索引、以太网帧的源MAC地址、源IP地址、目的IP地址、源端口、目的端口、协议、观测到首包时刻和观测到末包时刻。其中，观测到首包时刻和末包时刻的含义分别与NetFlow输出包的first switched和last switched相同。

在收集器端，溯源记录根据不同的发送路由器和入接口索引被分组，在每一组中，按时间顺序存储。

1.3.4 在覆盖网络中的溯源

在SampleTrace中，攻击流被逐跳向上游逻辑邻居溯源，每次将检查是否上游逻辑邻居采样了这个攻击流。

在溯源过程开始前，溯源发起实体E（受害者或者IDS）将会确定攻击流的五元组5-tuple：源IP地址、目的IP地址、源端口、目的端口和协议。SampleTrace对于实体E有一个限制：攻击流的五元组5-tuple必须被及时地确定。这是由于收集器存储空间的大小有限，在相应的溯源记录在收集器端被覆盖之前，溯源过程必须被启动。

在一个溯源请求中，P是已经被重构的部分攻击路径。假如是E发出的溯源请求，P被设置为null，当TS_i从E接收到一个溯源请求，TS_i将验证发送方的身份以及对溯源请求进行完整性检查。

在成功验证和检查后，TS_i执行SampleTrace溯源算法（参见图4）。在该溯源算法中，AS_j是AS_i的一个下游逻辑邻居，AS_m是AS_i的一个物理邻居，AS_k是AS_i的任意一个上游逻辑邻居；TS_i向TS_j发送HIT消息或者NO HIT消息表明是否五元组5-tuple命中了AS_i的收集器的溯源记录；当溯源失败时，TS_i向溯源发起实体E发送TRACEBACK FAIL消息。

由第1.3.3小节可知，每个存储在收集器的溯源记录有10个属性。因此，假如一个溯源记录与5-tuple匹配，那么就可以确定攻击流从AS_i的BGP路由器R_p的第ifindex个接口（即AS_i-R_p-ifindex）进入AS_i，并且也可以确定以太网帧的源MAC地址，即AS_i-R_p-ifindex的对等接口的MAC地址。根据第1.1小节的假设2），TS_i知道该MAC地址的接口属于物理邻居AS_m。因此，AS_m也可以随后被确定。也就是说，AS_m和AS_i-R_p-ifindex能被证明在攻击路径上。在下一步，根据AS_i的物理邻居AS_m是否是部署点AS，AS_i将发送不同数量的新请求以进行进一步的溯源。其中，P'包括AS_i-R_p-ifindex和P，不包括新请求的接受者所在的AS。

1) AS_m是部署点。TS_i将只发送一个新的溯源请求给TS_m，以查询该攻击流是从AS_m的哪个BGP路由器的哪个接口进入AS_m。新溯源请求中P'在SampleTrace溯源算法的第13行进行了赋值。如果AS_m没有采样攻击流，TS_m将发送NO HIT消息给TS_i，TS_i将终止溯源过程。

```
1  // 5-tuple 是攻击流的五元组，AS_m 为 AS_i 的一个物理邻居
2  //P 和 P' 是重构出的部分攻击路径
3  //E 是溯源发起实体的 IP 地址
4  //AS_i-R_p-ifindex 表示 AS_i 的 BGP 路由器 R_p 的接口 ifindex
5  //result 是溯源结果
6  假设 req(5-tuple, P, E)是 TS_j(或 E) 发送给 TS_i 的溯源请求;
7  TS_i 查询 AS_i 的收集器是否存储了 5-tuple 的溯源记录;
8  假如 AS_i 的收集器存储了 5-tuple 的溯源记录，则
9  可确认攻击流从 AS_m 通过 AS_i-R_p-ifindex 进入 AS_i;
10  向 TS_j(或 E)发送 HIT 消息;
11  result: ="AS_m, AS_i-R_p-ifindex,"+P;
12  假如 AS_m 是部署点，则
13    P': ="AS_i-R_p-ifindex,"+P;
14    TS_i 向 TS_m 发送 req(5-tuple, P', E);
15    等待 TS_m 的响应;
16    假如 TS_m 的响应为 NO HIT 消息，则
17      发送 result 给 E;
18  否则
19    假如 S_u(AS_i)为空或 S_u(AS_i)的成员均在 P 中，则
20      发送 result 给 E;
21    否则
22      P': ="AS_m, AS_i-R_p-ifindex,"+P;
23      对于 S_u(AS_i)中的每个成员 AS_k
24        假如 AS_k 不在 P 中，则
25          发送 req(5-tuple, P', E) 给 TS_k;
26      等待响应;
27      假如所有的响应均为 NO HIT 消息，则
28        发送 result 给 E;
29  否则
30  假如 req(5-tuple, P, E) 由 TS_j 发出，则
31    发送 NO HIT 消息给 TS_j;
32  假如 req(5-tuple, P, E) 由 E 发出，则
33    发送 TRACEBACK FAIL 消息给 E
```

图4 SampleTrace溯源算法

2) AS_m不是部署点。因为TS_i不能确定AS_i的哪个上游逻辑邻居将攻击流转发给AS_i，所以TS_i将向$S_u(AS_i)$中每个成员AS的溯源服务器发送一个溯源请求。其中，新溯源请求中P'在SampleTrace溯源算法的第22行进行了赋值。当下面3个条件中的任何一个满足时，TS_i将终止溯源过程：1) $S_u(AS_i)$为空；2) $S_u(AS_i)$中的所有AS成员在P中；3) $S_u(AS_i)$中没有成员采样或转发过该攻击流。

当TS_i终止溯源过程时，TS_i将发送溯源结果给溯源发起实体E。

图5是一个溯源过程的例子。每个溯源服务器通过上游逻辑邻居发现机制知道它所属AS的上游逻辑邻居。这样，覆盖网络建立起来了。当一个受害者被攻击后，攻击流将从受害者到攻击者的方向逐跳地被追踪。受害者发送一个溯源请求给TS_{10}，启动了溯源过程，如果TS_{10}采样了该攻击流，就能确定该攻击流通过MAC地址为MAC_6的接口和AS_{10}-R_3-5(AS_{10}-R_3-5表示AS_{10}的BGP路由器R_3的第5个接口)进入AS_{10}。根据第1.1小节的假设，TS_{10}知道MAC地址为MAC_6的接口属于AS_6并且AS_6为部署点AS，因此攻击流来自AS_6，TS_{10}只发送溯源请求给TS_6。如果AS_6采样了攻击流，就能确定攻击流通过MAC地址为MAC_8的接口和AS_6-R_2-3进入AS_6。因为TS_6知道MAC地址为MAC_8的接口属于AS_8并且AS_8是非部署点，所以攻击流来自于AS_8。TS_6给AS_6的每个上游逻辑邻居的溯源服务器(TS_3，TS_4，TS_5，排除TS_{10})发送一个溯源请求，TS_4和TS_5查询各自的收集器后，向TS_6发送NO HIT消息。如果AS_3曾采样到该攻击流，MAC地址为MAC_1的接口以及AS_3-R_1-1能够被确定，TS_3知道MAC地址为MAC_1的接口属于AS_1，因此攻击流来自于AS_1。因为AS_1是非部署点，AS_3的上游逻辑邻居是AS_6，但是AS_6已经被确认在攻击路径上，所以溯源过程在TS_3被终止，TS_3将重构出的攻击路径(AS_1，AS_3-R_1-1，AS_8，AS_6-R_2-3，AS_{10}-R_3-5)发送给受害者。由于采样的概率特性，有可能只有该攻击路径的后部分能够被成功重构。

2 性能评估

一个分布式拒绝服务(distributed denial of service，DDoS)攻击包括很多攻击流，在下面的讨论中，本文将分析其中任意一个攻击流被成功溯源的概率。基于对一个攻击流的溯源分析，可以对于DDoS攻击的溯源进行进一步分析。

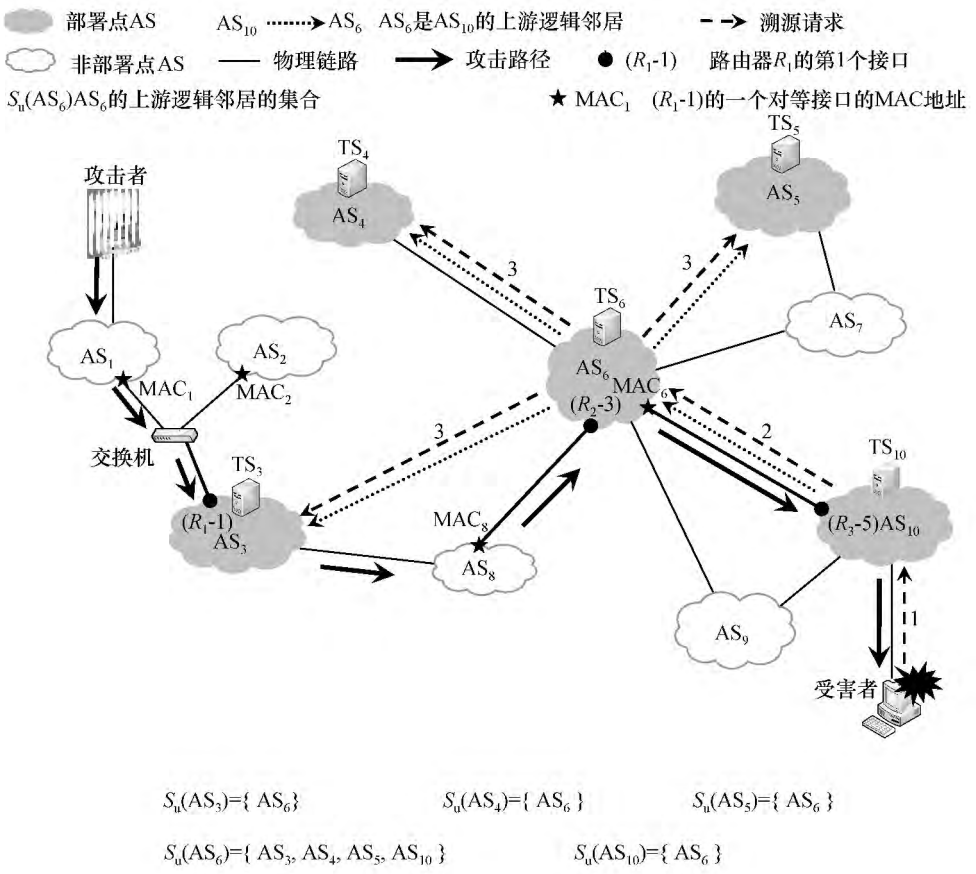

图 5 SampleTrace 溯源过程的例子

假定一个 m 包的攻击流经同一条攻击路径攻击受害者，这条攻击路径从攻击源到受害者依序经过 r 个($r\geqslant 1$)部署点 AS。假设这 r 个部署点 AS 对攻击流均进行独立概率采样，采样概率依序为 p_r，…，p_2，p_1。根据 SampleTrace 的溯源过程，在覆盖网络中，该攻击流的攻击路径被逐跳地从受害者到攻击源溯源。本文将按照两种不同的情况分析 SampleTrace 对该 m 包攻击流成功溯源 h 跳(AS 级，从受害者到攻击源方向)的概率 P_s。应该强调的是，根据 Bernoulli 大数定律[10]，当大量的 m 包攻击流在因特网中被 SampleTrace 溯源时，这些攻击流被成功溯源 h 跳的相对概率将接近于 P_s。

1) 第 h 个和第($h-1$)个部署点 AS 不直接相连，并且在两者之间不存在任何其他的部署点 AS(如图 6a 所示)。在这种情况下，假如 SampleTrace 能将该 m 包攻击流成功溯源 h 跳(AS 级，从受害者到攻击源方向)，则这 h 个部署点 AS 必须都采样到这个 m 包攻击流。因此，

$$P_s=\prod_{k=1}^{h}[1-(1-p_k)^m],\quad 1\leqslant h\leqslant r. \quad (2)$$

为了便于分析，假定每个部署点 AS 都采用相同的独立采样概率。图 7 是在独立采样概率分别为 1/20 和 1/100 的情况下，SampleTrace 对攻击流成功溯源不同跳数的概率与该流所包含的包数的关系。图 7 表明：1) 期望被成功溯源的跳数(AS 级)越多，该攻击流被成功溯源的概率越小；2) 独立采样概率越大或者攻击流所含的包数越多，该攻击流被成功溯源的概率越大。当部署点 AS 统一采用其他的独立采样概率时，曲线的变化趋势和图 7 相似。

2) 第 h 个部署点 AS 是第($h-1$)个部署点 AS 的物理邻居，也就是两者直接相连(如图 6b 所示)。假如从受害者到攻击源方向的($h-1$)个部署点 AS 都采样到了该攻击流，本文能够确定接口 MAC_q 和 R_i-j，R_i-j 为该攻击流进入第($h-1$)个部署点 AS 的入口点。并且，接口 MAC_q 属于第 h 个部署点 AS，因此即使第 h 个部署点 AS 没有采样到这个攻击流，第 h 个部署点 AS 依然能被确定在攻击路径上。

$$P_s=\prod_{k=1}^{h-1}[1-(1-p_k)^m],\quad 1\leqslant h\leqslant r. \quad (3)$$

式(2)和(3)的不同在于求积参数，一个是 h，另一个为 $h-1$。当根据式(3)作出示意图时，本文发现该

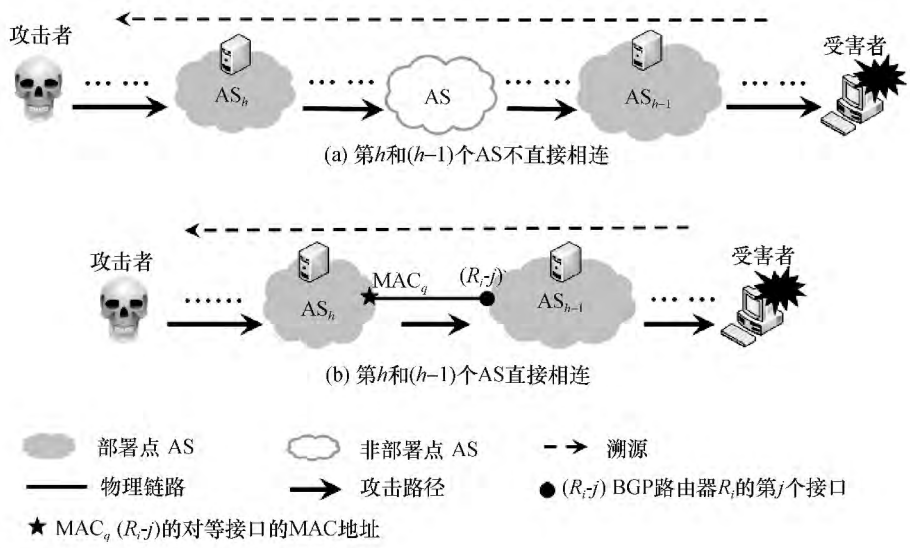

图 6 第(h−1)个和第 h 个部署点 AS 之间的两种连接关系

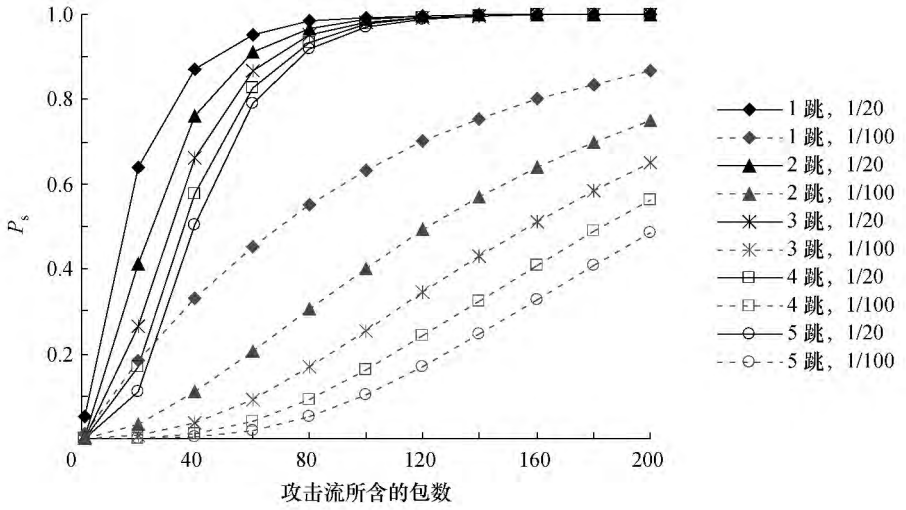

图 7 一个攻击流被成功溯源不同跳数(AS 级别)的概率(独立采样概率分别为 1/20,1/100)

图类似于图 7。

3 讨论

3.1 因特网地址溯源的功能

因特网地址溯源问题就是要确定直接产生攻击流量的机器和攻击流量经过的网络路径[11]。需要强调的是,攻击流量的直接产生者并不一定是真正的攻击者。例如,跳板攻击[12](stepping-stone attacks)、反射式攻击(reflected attacks)、通过隧道或者代理发动的攻击可以被分别溯源到跳板、反射器、隧道终点或者攻击代理。使用溯源方法追踪攻击流量的直接产生者不是确定真正攻击者的最后一步而是第一步,接着借助于跳板攻击检测技术或者网络安全领域的其他技术,可以确定真正的攻击者。

3.2 因特网地址溯源在应对僵尸网络中的作用

僵尸网络发动的 DDoS 攻击的攻击链是长而复杂的:一个僵尸主控机(botmaster)通过一些方式(例如:虚拟专网[13](virtual private network, VPN),代理或者跳板)控制命令和控制服务器(C&C server(s)),命令和控制服务器控制僵尸发送伪造源地址的攻击包攻击受害者。溯源方法可以确定僵尸和僵尸的攻击路径。但是,通过僵尸确定命令和控制服务器、通过命令和控制服务器确定僵尸主控机是两个不同于因特网地址溯源的独立问题。目前,针对前面两个问题,已经有了一些研究工作,例如加利福尼亚大学伯克利分校和加利福尼亚大学圣地亚哥分校的 Botfarm 项目[14-15]、北卡罗来纳州立大学的研究者发布的论文"A first step toward live botmaster traceback"[16]等。因特网地

址溯源不是用来解决这两个问题的，但可以被集成到追踪长而复杂的僵尸攻击链的完整解决方案中。

在大量的僵尸被溯源方法确定之后，有3种方式有助于减轻或者消除僵尸网络攻击的影响：

1）可以通知负责僵尸所在网络的管理员查杀僵尸软件、扫描计算机漏洞、打上软件补丁。

2）在溯源方法定位了僵尸后，缩小了僵尸的范围，有利于进一步地检测和分析僵尸网络。随后，可以采取一些措施应对僵尸网络攻击。例如：撤销被僵尸网络使用的恶意域名，定位和关闭命令和控制服务器（例如，对抗Waledac and Rustock僵尸网络的措施），或者其他可以摧毁命令和控制服务器与僵尸之间命令和控制通道的方式，包括DNS Sinkhole[17]。因此，因特网地址溯源有助于阻止或者消除僵尸网络发动的DDoS攻击。

3）因特网地址溯源除了可以确定攻击流量的产生者之外，还可以确定攻击流量的路径。因特网地址溯源收集的信息可以帮助攻击流量汇聚的因特网服务提供商（Internet service provider，ISP）来应对DDoS攻击：

a）假如这些ISP部署有DDoS攻击应对系统（例如，流量清洗系统），因特网地址溯源可以提供攻击流量的特征信息给这些DDoS攻击应对系统，这有助于减轻或者消除DDoS攻击的危害。

b）因特网地址溯源可以提供攻击流量的特征信息给ISP，ISP可以在相关的路由器上采取过滤或者限流措施来应对DDoS攻击，这可以减轻或者消除受害者所遭受DDoS攻击的强度。

总之，因特网地址溯源不能（也没有其他方法可以）独立阻止僵尸网络发动的DDoS攻击，但是因特网地址溯源有利于追踪长而复杂的僵尸网络攻击链。并且，因特网地址溯源收集的信息能帮助ISP/网络管理员来定位、消除僵尸和攻击包，因此可以减轻或者消除僵尸网络发动的DDoS攻击的强度。

3.3 采样概率

在第2节提到，SampleTrace的性能与采样概率相关。当其他参数保持不变时，采样概率越高，溯源的性能越好。但同时，对于支持xFlow的BGP路由器，采样概率越高，开销越大。为了处理这个问题，可以采用图8所示的采样方案，该采样方案不会对BGP路由器有任何的影响，并且支持更高的采样率（例如，1，1/5，1/10，1/20等）。在图8中，在BGP路由器的外接链路上设置网络分流器[18]（例如，分光器），网络分流器无阻碍地转发A和B之间的流量，同时将流量拷贝传送给C，C连接流量分割仪[19]。流量分割仪可以将输入的高速流量分割为多个低速流量输出，例如，输入到10 Gb口的流量可

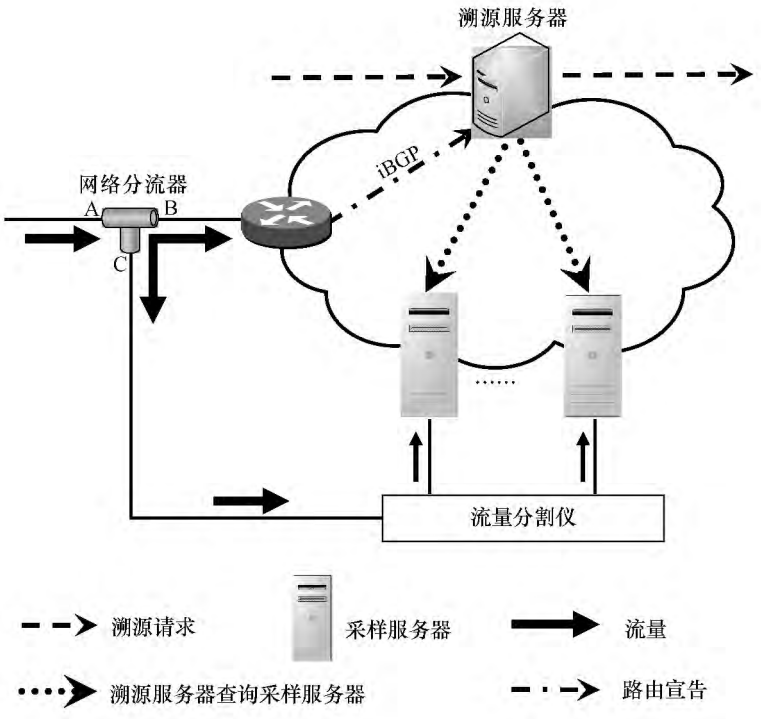

图8 使用网络分流器和流量分割仪的采样方法

以被分割,从 10 个 1 Gb 口输出[20]。因此,SampleTrace 支持更高的采样概率;该采样机制可以使得多个低性能的采样服务器有能力对高速流量进行采样,并且聚合采样的数据包为流粒度的信息,以溯源记录的形式存储在本地上。在溯源服务器与采样服务器之间的通信机制类似于溯源服务器与收集器之间的通信机制。并且,将溯源服务器通过专用网络与采样服务器相连也是可行的。

4 结束语

对于绝大多数因特网地址溯源方案,部署的困难是非常有挑战性的问题。本文基于现有的协议和功能(例如 BGP,sFlow,NetFlow 和 IPFIX)提出的 SampleTrace 方案,不仅在域内还是域间均可增量部署。SampleTrace 不同于现有的日志记录类溯源方案,可以向受害者或者 IDS 提供 AS 级别的溯源方案,并且 SampleTrace 可以确定攻击流进入部署点 AS 时的入口路由器和相应的接口。SampleTrace 可概率性地发现攻击源 AS 和攻击流量的路径。本文也提出了 SampleTrace 的部署激励。将来,本文作者将优化收集器的存储开销,并且在 CNGI-CERNET2[21] 上作部署实验。

参考文献 (References)

[1] Phaal P, Lavine M. sFlow Version 5 [EB/OL]. [2012-08-01]. http://www.sflow.org/sflow_version_5.txt.

[2] Cisco Systems. NetFlow Services Solutions Guide [EB/OL]. [2012-08-01]. http://www.cisco.com/en/US/docs/ios/solutions_docs/netflow/nfwhite.html#wp1030098.

[3] RFC 3954. Cisco Systems NetFlow Services Export Version 9 [S]. Claise B, Li T, Hares S. Prague: IETF Network Working Group, 2004.

[4] RFC 6728. Configuration Data Model for the IP Flow Information Export (IPFIX) and Packet Sampling (PSAMP) Protocols [S]. Muenz G, Claise B, Aitken P. Paris: IETF IPFIX Working Group, 2012.

[5] RFC 6526. IP Flow Information Export (IPFIX) per Stream Control Transmission Protocol (SCTP) Stream [S]. Claise B, Aitken P, Johnson A, et al. Paris: IETF IPFIX Working Group, 2012.

[6] RFC 6313. Export of Structured Data in IP Flow Information Export (IPFIX) [S]. Claise B, Dhandapani G, Aitken P, et al. Paris: IETF IPFIX Working Group, 2012.

[7] RFC 5101. Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information [S]. Claise B. Vancouver: IETF IPFIX Working Group, 2008.

[8] RFC 5103. Bidirectional Flow Export Using IP Flow Information Export (IPFIX) [S]. Trammell B, Boschi E. Vancouver: IETF IPFIX Working Group, 2008.

[9] RFC 4271. A Border Gateway Protocol 4 (BGP-4) [S]. Rekhter Y, Li T, Hares S. San Diego: IETF Network Working Group, 2006.

[10] Kallenberg O. Foundations of Modern Probability [M]. 2nd Ed. Berlin: Springer-Verlag, 2006.

[11] Savage S, Wetherall D, Karlin A R, et al. Practical network support for IP traceback [C]// Proc ACM SIGCOMM. New York: ACM Press, 2000: 295-306.

[12] Zhang Y, Paxson V. Detecting Stepping Stones [EB/OL]. [2012-10-11]. http://www.cs.jhu.edu/ ~ fabian/courses/CS600.424/course_papers/Stepping-Stones.pdf.

[13] Wikipedia. Virtual Private Network [EB/OL]. [2012-10-11]. http://en.wikipedia.org/wiki/Virtual_private_network.

[14] Cho C Y, Babic D, Shin C E R, et al. Inference and analysis of formal models of botnet command and control protocols [C]// Proc ACM Conf on Computer and Communications Security. New York: ACM Press, 2010: 426-439.

[15] Caballero J, Poosankam P, Kreibich C, et al. Dispatcher: Enabling active botnet infiltration using automatic protocol reverse-engineering [C]// Proc ACM Conf on Computer and Communications Security. New York: ACM Press, 2009: 621-634.

[16] Ramsbrock D, Wang X Y, Jiang X X. A first step toward live botmaster traceback [C]// Proc RAID. Washington DC: IEEE Computer Society, 2008: 59-77.

[17] Wikipedia. DNS Sinkhole [EB/OL]. [2012-10-11]. http://en.wikipedia.org/wiki/DNS_Sinkhole.

[18] Wikipedia. Network Tap [EB/OL]. [2012-10-11]. http://en.wikipedia.org/wiki/Network_tap.

[19] Shanghai EmbedWay Information Technologies Products [EB/OL]. [2012-10-11]. http://www.embedway.com/en/products.php.

[20] Shanghai EmbedWay Information Technologies EmbedWay Products [EB/OL]. [2012-10-11]. http://www.embedway.com/en/product_view.php?id=2.html.

[21] CNGI-CERNET2. China Next Generation Internet Project: China Education and Research Network (CERNET) [EB/OL]. [2014-06-10]. http://www.cernet2.edu.cn/index_en.htm.