ISSN 1673-9418 CODEN JKYTA8
Journal of Frontiers of Computer Science and Technology
1673-9418/2016/10(05)-0601-11
doi: 10.3778/j.issn.1673-9418.1507049

E-mail: fcst@vip.163.com
http://www.ceaj.org
Tel: +86-10-89056056

使用Spark Streaming的自适应实时DDoS检测和防御技术*

方 峰,蔡志平+,肇启佳,林加润,朱 明
国防科学技术大学 计算机学院,长沙 410073

Adaptive Technique for Real-Time DDoS Detection and Defense Using Spark Streaming*

FANG Feng, CAI Zhiping+, ZHAO Qijia, LIN Jiarun, ZHU Ming
College of Computer Science, National University of Defense Technology, Changsha 410073, China
+ Corresponding author: E-mail: zpcai@nudt.edu.cn

FANG Feng, CAI Zhiping, ZHAO Qijia, et al. Adaptive technique for real-time DDoS detection and defense using Spark Streaming. Journal of Frontiers of Computer Science and Technology, 2016, 10(5): 601-611.

Abstract: Distributed denial of service (DDoS) attack is an important security threat, the constant improvement of network speed to the traditional detection methods has brought new challenges. Represented by Spark and so on, the big data processing technology brings new opportunity to the completion of high-speed safety detection. This paper proposes an adaptive technique for real-time DDoS detection and defense using Spark Streaming framework. Based on source cluster grouping in sliding windows, and the deviation of proportionate to the source cluster of groups, this paper detects out the DDoS attack traffic trace, and realizes the adaptive rapid and precise detection of DDoS attacks through sensing legitimate network traffic. The experimental results show that the technique can greatly improve detection capabilities, in order to ensure the security of network service performance and the extensibility of detection, this paper provides a feasible solution.

Key words: DDoS detection; DDoS defense; real-time detection; adaptive detection; Spark Streaming

摘 要:分布式拒绝服务(distributed denial of service,DDoS)攻击是重要的安全威胁,网络速度的不断提高给传统的检测方法带来了新的挑战。以Spark等为代表的大数据处理技术,给网络安全的高速检测带来了新的

* The National Natural Science Foundation of China under Grant Nos. 61379145, 61363071 (国家自然科学基金).
Received 2015-07, Accepted 2015-09.
CNKI网络优先出版: 2015-10-16, http://www.cnki.net/kcms/detail/11.5602.TP.20151016.1108.006.html

契机。提出了一种基于Spark Streaming框架的自适应实时DDoS检测防御技术,通过对滑动窗口内源簇进行分组,并根据与各分组内源簇比例的偏差统计,检测出DDoS攻击流量。通过感知合法的网络流量,实现了对DDoS攻击的自适应快速检测和有效响应。实验结果表明,该技术可极大地提升检测能力,为保障网络服务性能和安全检测的可扩展性提供了一种可行的解决方案。

关键词:DDoS检测;DDoS防御;实时检测;自适应检测;Spark Streaming

文献标志码:A **中图分类号**:TP393

1 引言

分布式拒绝服务(distributed denial of service, DDoS)攻击是威胁极大的一类网络攻击行为。DDoS防护服务市场领导者Black Lotus发布的最新报告显示,全球大型服务提供商都饱受各种DDoS攻击。DDoS攻击范围非常广泛,涵盖各行各业,其中64%的平台提供商、66%的托管解决方案提供商和66%的VoIP服务提供商受到影响。DDoS攻击是用很多计算机发起协作性的DoS攻击来攻击一个或者多个目标,它试图耗尽一台服务器的计算资源,以使服务器无法响应合法用户的资源请求。其典型的攻击方法就是数据包泛洪,通过每秒发送数千个数据包到目标主机以致网络链路拥塞。大量的非法网络流量一般是黑客通过控制连接到网络的一大批肉机(僵尸网络)产生的。因为攻击规模大和不断变化,所以DDoS攻击检测和防御技术具有很大的挑战性。成功的防御机制,必须能有效地过滤掉恶意流量,且尽可能小地减少对合法用户流量的影响,能持续快速地对威胁做出响应,并具有微小的延时开销(特别是当没有遭到攻击时)。

实时异常检测[1]的目标是能实时捕获系统中的异常行为,包括接受连续不断的数据流,实时分析数据和及时的应对策略,它的挑战来自于输入数据的体积、流速和复杂多样性。但近几年,DDoS攻击呈现攻击流量不断增大的趋势,可达每秒数十GB甚至数百GB的攻击带宽,传统的防御技术和机制已很难应对。一旦防御失败,对于商业的持续运行将造成十分严重的后果。因此,企业数据中心的DDoS检测必须建立在低延时的有大数据处理能力的可扩展的框架上。

数据的迅猛增加使得Hadoop、Spark等大数据处理技术得到了快速发展。但是Hadoop目前只能处理批数据,无法处理实时数据。Spark是一个分布式的实时数据处理的优秀框架,相比Hadoop,它是基于内存的计算框架,避免了传统的MapReduce[2]编程模型带来的巨大的I/O通信开销瓶颈。目前的大数据处理性能在内存中可达到Hadoop的100倍,在硬盘中的速度也可达到Hadoop的10倍。

Spark有两个关键概念:弹性分布数据集(resilient distributed dataset,RDD)和有向无环图(directed acyclic graph,DAG)执行引擎。RDD是一个分布式的内存抽象,它允许在大型分布式集群上进行高容错的内存计算。Spark有两种RDD:基于现有编程集合(如map、list等)的并行集合和存储在HDFS中的文件。对RDD的操作分两种:转换和动作。转换是为输入的RDD或现存的RDD创建出一个新的数据集,动作是在执行对数据集的计算后返回一个值。相比而言,转换只是定义一个新的RDD,是一个惰性操作,而动作执行真正的计算,它能计算出结果或写入外部存储介质。每当用户对RDD进行动作,会在考虑所有转换的依赖关系后生成一个有向无环图,它消除了传统MapReduce的多步执行模型且提升了性能。

Spark也有对流的实现,具有高可扩展和高容错的特点。Spark Streaming是将流式计算分解成一系列短小的批处理作业。这里的批处理引擎是Spark,也就是把Spark Streaming的输入数据按照batch size(如1 s)分成一段一段的数据(discretized stream,DStream),每一段数据都转换成Spark中的RDD,然后将Spark Streaming中对DStream的Transformation操作变为Spark中对RDD的Transformation操作,将RDD经过操作变成中间结果保存在内存中。整个流

式计算根据业务的需求可以对中间结果进行叠加，或者存储到外部设备。

将数据流技术用于缓解DDoS攻击是网络安全研究的重要趋势，特别是最近已经出现了一些优秀的弹性可扩展的数据流引擎和框架[3]。本文做了以下工作：

（1）利用弹性可扩展的数据流框架Spark Streaming和Kafka消息中间件，实现了对连续不断到来的网络流量数据进行实时查询分析处理。

（2）利用基于源簇统计的分组策略，将正常流量与DDoS流量的偏差根据动态自适应计算出的流量阀值区分开，能有效检测出DDoS攻击，并制定针对攻击流量的缓解策略，有效防御应对DDoS攻击。

（3）在低延时和准确性两个方面对实时DDoS检测框架通过实验进行了验证。

2　相关工作

近年来，针对有效解决DDoS攻击问题的迫切需求，DDoS检测已成为研究热点。在网络入侵方面主要有两类检测方法[4]：基于特征的方法和基于异常的方法。基于特征的方法[5]通过检查每一个数据包的特征，以决定转发或丢弃。但是它有一些局限性：不是所有的协议都可以被标记出特征，并且针对每个新型的攻击，需要有与之对应的特征才能被检测出[6]。与基于特征的方法相反，基于异常的方法试图通过覆盖更宽的范围，从当前和相关的流量中[4]发现偏差以检测威胁。因为单独来看，每一个恶意数据包可能是合法的，但通过识别当前和上下文的流行为后，就会发现它是恶意的。很多基于异常的方法，通过对流量特征的复杂分析[6-11]，可检测和缓解多种类型的攻击。目前的挑战在于如何定义一种适合在线实时处理，并且能足够精确缓解攻击的流量分析方法。文献[6]中提到的基于挖掘工具的方法更适合于用来做研究，而不适合于用来检测威胁。简单观察整体流量[12-15]的方法无法给出要丢弃哪些攻击数据包。

可以应用于网络安全检测的大数据处理技术包括MapReduce、Hadoop和Spark等，由于网络流量数据的大容量、实时性和不稳定性，现有的大数据技术框架可能无法稳定地进行处理。Linkedin开源的Kafka为消息的顺序传递提供了可靠的机制，是一个高吞吐量分布式消息系统中间件。首先，Kafka的开发者们认为不需要在内存里缓存什么数据，操作系统的文件缓存已经足够完善和强大，只要你不进行随机写，顺序读写的性能是非常高效的。Kafka的数据只会顺序append，数据的删除策略累积到一定程度或者超过一定时间再删除。Kafka另一个独特的地方是将消费者信息保存在客户端而不是MQ服务器，这样服务器就不用记录消息的投递过程，每个客户端都知道自己下一次应该从什么地方什么位置读取消息，消息的投递过程也是采用客户端主动pull的模型，这样大大减轻了服务器的负担。Kafka还强调减少数据的序列化和拷贝开销，会将一些消息组织成Message Set进行批量存储和发送，并且客户端在pull数据的时候，尽量以zero-copy的方式传输，利用sendfile这样的高级IO函数来减少拷贝开销。Spark Streaming 1.3.0版本提供了相关的API实现对Kafka的较好支持，因此可以实现对连续不断到来的网络流量数据进行实时查询分析处理。

3　自适应DDoS攻击检测防御架构

本文基于对网络DDoS攻击流量在给定时间窗口内的源簇特征的统计分析，提出了一种自适应的DDoS攻击检测防御架构。以下将介绍网络和流量模型，定义攻击模型和状态，对DDoS攻击检测和防御问题进行建模。

3.1　网络DDoS攻击流量模型

该网络包括4种类型的实体：（1）受保护的网络实体，即被攻击的网络主机；（2）合法主机，即占用受保护网络实体资源的终端；（3）DDoS防御系统主机，用于运行DDoS防御系统的主机；（4）僵尸机，即被攻击者控制的网络主机。

受保护的网络实体、合法主机和僵尸机通过网络链路和路由交换机相连，当通信链路中含有DDoS攻击流量时，其流量模型如图1所示。

通常DDoS攻击流量的特征是在较短的时间内发送大量的网络数据包，为了及时检测出DDoS攻击

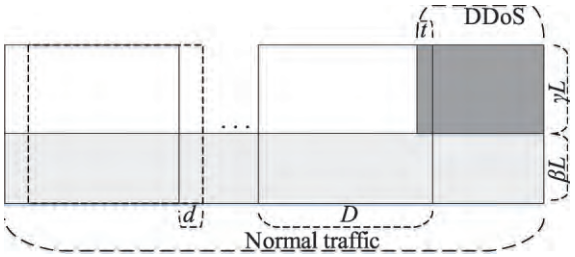


Fig.1 Traffic model of network DDoS attack

图1 网络DDoS攻击流量模型

流量，本文采用滑动时间窗口的检测方法。假设滑动窗口大小为D，滑动距离为d，网络最大流量负载为L，合法流量的平均负载为βL，攻击流量平均负载为γL。当滑动窗口内含有t时间长度攻击流量时，$t\times\gamma L$部分的流量源簇（按不同源IP地址统计的平均包数、平均包大小和平均包时间间隔）与$D\times\beta L$部分的流量源簇有一定偏差，因此，如果计算出偏差超出一定的阀值（合法流量源簇之间偏差最大值），就能检测出DDoS攻击。

本文定义了两个输入数据流：网络流S和聚合的网络流Sa。S表示发送到受保护网络实体的数据包流量，其中每个包可以被看作是一个元组<*srcIP, bytes*>，属性*srcIP*和*bytes*分别表示源IP地址和包的大小。Sa元组由S数据包信息在每个时间周期内以源IP地址聚合得到，其属性构成<*srcIP, tA, tB, packets, bytes*>（例如，给定时间段8:00:00—8:00:30，元组<*A*, 8:00:12, 8:00:25, 5, 250>表示IP地址*A*在8:00:12到8:00:25时间段内发送5个总大小为250 Byte的数据包）。聚合流Sa可以通过监测应用程序从网络流S中获得，如Cisco的Netflow协议，是网络设备和互联网服务供应商都广泛支持的协议。在数据流中，采用滑动时间窗口的方式对到来的数据流进行周期持续地采样。例如，窗口大小为1 h和滑动距离为10 min的时间窗口，将覆盖周期[8:00:00，9:00:00)，[8:10:00；9:10:00)，以此类推。

3.2 自适应DDoS攻击检测和防御框架

基于对3.1节网络DDoS攻击流量模型的分析，本文设计了自适应的DDoS攻击防御系统（图2）。检测控制中心（detection control center，DCC）根据Sa中当前时间窗口源簇的特征与历史数据集（historical dataset，HD）中源簇的参考特征比较，进行DDoS攻击检测。缓解中心（mitigation center，MC）设在输入流S和受保护的网络实体之间，当流量负载超过αL时就对流量进行过滤，过滤标准由DCC确定，MC的输出流Sm是S的子集。若流量负载低于αL，则MC的输出流Sm等于S，MC不被激活，只简单地转发S数据包，因为这些流量对受保护的网络实体来说开销很微小。DDoS攻击防御框架将计算并聚合多个相同源IP地址的流量特征，并以源簇（*srcCL*）形式被保存在各个分组中。

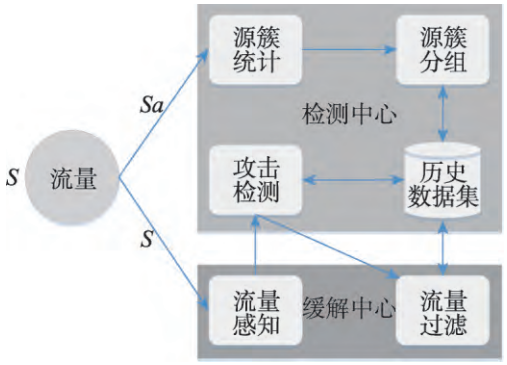


Fig.2 Adaptive DDoS attack detection and defense system architecture

图2 自适应DDoS攻击检测和防御系统架构

给定一个受保护的实体和它的最大负载L，通过监测超过负载阀值αL时的流量检测可能的威胁。每当应用过滤时，必须保证最大程度地将合法用户流量转发到受保护的网络实体。不仅要在真正有威胁时保护主机，也要在出现合法的峰值负载（如瞬间拥塞）时保护主机。因此，在攻击流量和由合法用户产生的峰值流量之前，使用真实的合法流量与受害主机保持通信。防御机制必须确保流入到受保护网络实体的恶意IPs在被检测出之前，无法达到网络的最大负载。其中的挑战在于在S中使用什么样的丢包标准，保证整体流量负载不超过L的前提下，尽可能多地转发合法流量。为什么只有当流量超过αL时才过滤流量，有两方面原因：一方面，本文的解决方案并不打算分析异常原因，也不区分流量峰值是否合法；另一方面，当没有超出αL时，转发潜在的恶意流量就使得攻击者很难适配系统对攻击的响应机制。

3.2.1　源簇分组策略

为了将DDoS攻击流量从混合流量中区分出来，采用了分组策略（图3），在分组前，先对D时间窗口内的流量按源地址聚簇（算法1），简称源簇（SourceIPs cluster, $srcCL_i$）。统计出相同源IP地址的平均特征$f_i=(\varphi_i, \omega_i, \tau_i)$，其中$\varphi_i$表示每条流的平均包数目，$\omega_i$表示每条流的平均包字节数，$\tau_i$表示每条流中的平均包间隔时间。算法1的第1~8行统计得到每个源地址的每条流的平均特征，第9~10行统计出并保存每个源地址的平均特征。因为DDoS攻击数据包在一段时间内都会呈现出比较固定的特征，如固定的每秒发包数目、固定的包字节、固定的平均间隔时间。取参考点$O=(O_\varphi, O_\omega, O_\tau)$（取特征的0.95分位点，因为在之前的研究[4]中表明，超过90%的网络数据流都是包数较少的微型流），即可将DDoS攻击流量集中分到某个分组中。

算法1　聚簇

```
1. Procedure CLUSTERTING
2.     srcCL ← null  /* 源簇集合初始化 */
3.     for each tuple t do  /* 遍历元组 */
4.         srcIP ← t.srcIP  /* 赋值源 IP 地址 */
5.         avgFeature ← {0, 0, 0}  /* 初始化每个源 IP
           地址对应的平均特征 */
6.         for each tuple tu do  /* 遍历元组 */
7.             if tu.srcIP equals srcIP then  /* 匹配相
               同的源 IP 地址 */
8.             GetFeature(tu.dstIP, tu.pktNum,
               tu.pktSize, tu.pktInterval) /* 特征统计 */
9.         avgFeature ← {avgPktNum, avgPktSize,
           avgPktInterval}  /* 赋值统计出的平均特征 */
10.        srcCL.addFeature(srcIP, avgFeature)  /* 插
           入源簇集合 */
```

3.2.2　检测控制中心

检测控制中心（DCC）接收流量Sa，属于同一源簇的源IP地址特征结合在一起，因此每个源簇i（$srcCL_i$）由特征$f_i=(\varphi_i, \omega_i, \tau_i)$表示。其中$\varphi_i$表示每条流的平均包数目，$\omega_i$表示每条流的平均包字节数，$\tau_i$表示每条流中的平均包间隔时间。在当前窗口中的信息是指最近的数据部分，例如，φ_i可以表示为在刚过去一小时内，从$srcCL_i$发送到受保护网络实体的每条流的平均包数目。通过将源IP地址簇进行分组观察源簇的走向，并研究各个分组中源簇数比例随时间变化的情况。具体而言，将空间划分成8个不同的组$\{G_0, G_1,\cdots,G_7\}$，并保存属于各组的源簇数目。通过比较源簇特征和参考点$O=(O_\varphi, O_\omega, O_\tau)$进行分组（图3）。参考点不是固定值，它必须根据流入受保护网络实体的流量特征计算得到，其值取保存在历史数据集中的每个特征值的0.95分位数。之所以是0.95分位数，是因为在之前的研究[4]中表明，超过90%的网络数据流都是包数较少的微型流。本文的这种方法，将大部分（如果3个特征之间相互独立，则95%×95%×95%≈85%）流向受保护网络实体的流（在短时间周期被发送的数目较少和字节较短的包）分到了G_0组。

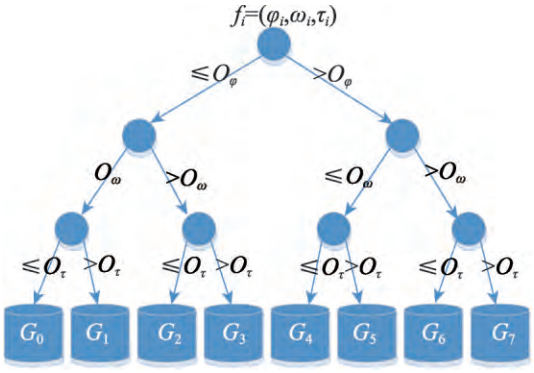


Fig.3　Grouping strategy

图3　分组策略

如果新的源簇的特征预分组的数目比例与相应每个分组的数目比例的偏差超出了阈值，则认为有异常。当前流量中，属于各分组的源簇数目用$\{\hat{n}_0, \hat{n}_1,\cdots,\hat{n}_7\}$表示，相对应的各分组中的参考源簇数目用$\{n_0,n_1,\cdots,n_7\}$表示。使用这些量计算每个分组的当前比例$\hat{r}_i=\hat{n}_i\big/\sum_j\hat{n}_j$和参考比例$r_i=n_i\big/\sum_j n_j$。如果参考比例与当前比例之间的差值超出给定阈值$tol$，将检测出流量异常，例如，$\max_i|r_i-\hat{r}_i|\geqslant tol$。检测偏差阈值$tol$受合法流量之间统计量的最大偏差所约束，若$tol$过小，则造成较高的误检测率，若$tol$过大，则检测出攻击流量的延迟较大。因此在使用下文所述的检测方法前，对已设定了滑动时间窗口D和滑动时间

距离d的合法流量之间统计量的偏差范围进行了不间断的评估,以动态确定出合适的检测偏差阀值tol。

参考点O和参考比例$r_0,r_1,\cdots,r_7$,这些值基于保存在历史数据集中的过去D时间段内(例如,星期三11:00:00—12:00:00)的流量特征信息计算得到。参考比例r_i由$\sum_d n_i^{(d)} \Big/ \sum_{j,d} n_j^{(d)}$计算得到,其中$n_i^{(d)}$是第$i$分组在$d$时间间隔的源簇数目。参考点$O$由在前面$D$时间段内流入受保护网络实体的每个源簇特征的加权0.95分位点计算得到。源簇对O的贡献为其在相关周期时间内出现数目的比例。因此,为每个源簇$srcCL_i$定义了表示在D时间段内出现数目的W_i。源簇$srcCL_i$的特征$f_i=(\varphi_i, \omega_i, \tau_i)$最后一个$D$时间段内的特征的平均值计算得到。所有的源簇$srcCL_i$就都有一个权重$W_i$和一个平均特征$f_i$,特征$\varphi$的加权0.95分位点由排序后的所有源簇平均特征$\{\varphi_i\}_i$在带有权重的$0.95\sum_i w_i$位置计算得到。例如,当一次数据包泛洪攻击引入大量新源簇时,持续监视受保护网络的流量,一旦源簇在各分组的分布发生突变,就检测出了异常。

3.2.3 缓解中心

缓解中心(MC)能过滤掉恶意流量并极小影响合法流量而缓解攻击。算法2给出了流向受保护网络实体的数据包是否被丢弃的步骤和顺序。第1~4行,若没有超出受保护网络实体的最大负载且缓解没被激活,则数据包将被转发。第5~6行,若减缓被激活,缓解中心必须保证让合法流量优先转发的同时不超过网络的最大负载。属于G_0组流量的过滤优先让在检测威胁之前与受保护网络实体的源簇通过。第7~9行,因为大部分源簇都属于G_0组,所以属G_0组的流量用布隆过滤器(bloom filter,BF)[16]过滤。另一方面,属$G_1, G_2, \cdots, G_7$组流量的过滤优先让频繁与受保护网络实体通信的源簇通过,在这种情况下,第10~11行,根据保存在白名单(acquaintance-list,AL)里的信息对每个源簇进行过滤。此外,属于每个分组包的转发还要由它将占用多少链路的负载所决定,因此每个分组的流量尽可能让合法流量优先通过。第12~13行,那些分组未知的源簇(可能既不是攻击也不是瞬间拥塞的流量)的数据包的转发要依赖于是否有可用的负载。第14~15行,过滤掉不属于以上条件的流量。

算法2 过滤

```
1. Procedure FILTERING(InputStream)
2.     srcCL ← CreateCluster(InputStream) /* 从输入流创建源簇 */
3.     if exceed max load L then /* 若当前流量超出网络最大负载 */
4.         Discard(InputStream) /* 进行过滤 */
5.     else if mitigation is not active then /* 若缓解没被激活 */
6.         GoToEntity(InputStream) /* 转发当前流量至受保护网络实体 */
7.     else
8.         if BelongToG0(srcCLi)
               &InBloomFilter(srcCLi)
               &hasAvailLoad then
9.             GoToEntity(srcCLi, srcIP) /* 若源簇属G0组,在布隆过滤集合中,有可用负载,则转发 */
10.        if BelongToOtherGroup(srcCLi)
               & InAcquainList(srcCLi)
               & hasAvailLoad then
11.            GoToEntity(srcCLi, srcIP) /* 若源簇属其他组,在白名单集合中,有可用负载,则转发 */
12.        if BelongToUnKnown(srcCli)
               & hasAvailLoad then
13.            GoToEntity(srcCLi, srcIP) /* 若源簇分组未知,有可用负载,则转发 */
14.        else
15.            DisCard(srcCLi, srcIP) /* 若其他情况,被过滤 */
```

布隆过滤器(BF)用来过滤属于G_0组源簇的数据包,这是一个空间有效的概率性数据结构,被用于检查一个集合中是否有某个元素[16]。布隆过滤器允许新的元素加入到集合里,但元素的删除并不是微不足道。本文将基本的布隆过滤器改进为一个基于时间戳的布隆过滤器,属于集合的元素与时间戳相关联,这个时间戳在被加到布隆过滤器时就被指定(基

于时间的BF跟基础的BF[16]具有相同的时间空间上的复杂性）。这样，能保存属于G_0组一定时间周期内的源簇（本文缓解机制里的最后5 min）。

白名单（AL）用于保存那些属于G_1, G_2,…,G_7组的源簇。当缓解攻击时，每一个源簇会被指定一个被转发的概率和比例。比起偶尔与受保护网络实体通信的源簇数据包，那些属于与受保护网络实体频繁通信的源簇的数据包将有更高的概率被转发。

应该注意的是，无论是使用BF还是AL，人们始终不能确保流向受保护网络实体的单一源簇不会超出链路的最大负载L。虽然攻击者不了解受保护网络实体的流量，但如果一组源地址与保存在BF和AL的有重叠，就需要一个机制确保这些流量不会让受保护网络实体的容量饱和。出于这个目的，根据每个分组平时正常的负载情况，为每个分组分配一个相对于最大负载的一个负载比例。

4　基于Spark Streaming的DDoS攻击检测和防御架构

本文的DDoS攻击防御框架使用了VMware集群配置，集群由两台浪潮英信服务器组成，每台具体配置是英特尔Xeon® CPU E7-4820 v2 2.00 GHz处理器，32 GB DDR3内存，4 TB硬盘，每台处理器有18个核，32个线程，安装Linux CentOS6.5 64位系统。集群包括3个虚拟机，每个虚拟机安装JDK/JRE v1.7版本，CDH v5.0发行版本（内含Spark v1.3.0版本和Hadoop v2.4.0版本），所有虚拟机共享硬件资源，一个控制节点，两个计算节点。

本文的技术架构（图4）主要由两个组件构成，一个是消息中间件（Kafka）模块，另一个是实时统计检测的Spark集群。

4.1　消息中间件模块

本文通过消息中间件将Netflow协议产生的流量日志文件与Spark集群相连接，Spark可集成一些消息中间件（如Apache Kafka、RabbitMQ等），之所以选择Kafka v3.3.4版本，是因为它可非常稳定地与Spark兼容。Kafka为消息传输创建了专用的队列，以正确的顺序提供有保障的消息传输，人们可以使用Kafka集群处理大体积的数据。

4.2　实时检测的Spark集群

如图4所示，实现了一个Spark集群，流量数据以连续不断的流通过Kafka队列传给Spark集群处理。流量被分割成微小的批次（DStream），对DStream进行转换和相关的动作，通过算法3检测出DDoS攻击。第2~4行对流进行Map操作后将二进制流处理成字符串元组，再把元组Map成新的DStream。第5

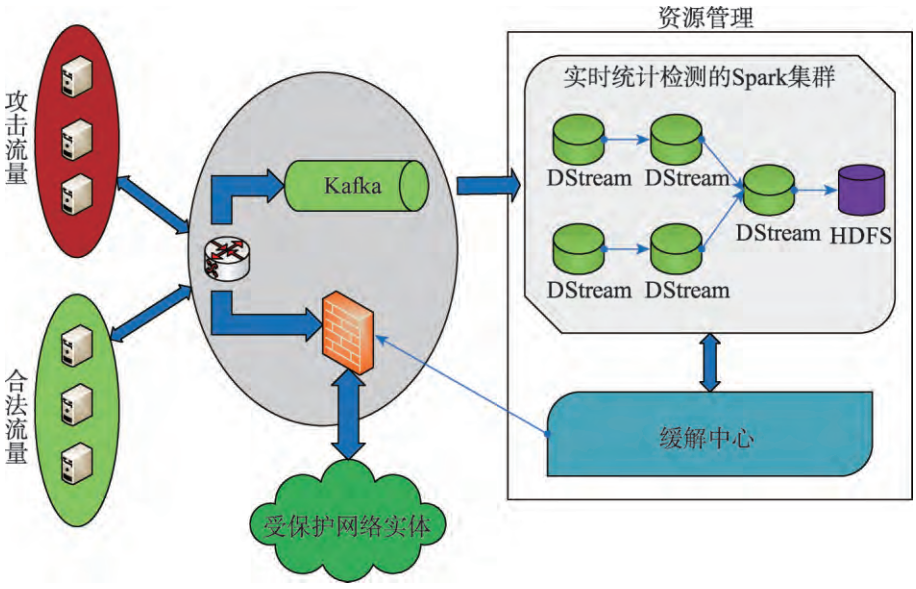


Fig.4　DDoS attack detection and defense architecture based on Spark Streaming

图4　基于Spark Streaming的DDoS攻击检测和防御架构

行使用了 Spark Streaming 框架支持的滑动窗口对 DStream 进行处理。第 6 行对新窗口内的不同源 IP 地址进行聚簇。第 7~9 行根据源簇的分组比例计算出窗口内的统计偏差。第 10~16 行与检测阀值 *threshold* 比较,通过作用于缓解中心模块,若未超出阀值,则不过滤当前流量,直接将其流入受保护网络实体,若超出阀值,则过滤当前流量,有选择地将其流入受保护网络实体。

算法 3 检测新窗口数据流异常

```
1. Procedure DETECTING(InputDStream)
2.    dtModel ← InputDStream.map(Stream s){  /* 对输入流量数据进行 Map 操作*/
3.        tuple ← ParseTuple(s)  /* 从流量数据中解析出元组对象 */
4.        return Map(tuple.srcIP, tuple)  /* 返回以源IP地址为 key,与之对应元组对象为 value 的键值对并进行 Map操作*/
5.    }.reduceByKeyAndWindow(srcIP, [Tuple], windowLength, slidingInterval){  /* 设定数据处理的时间窗口和滑动距离 */
6.        srcCL ← CreateCluster(srcIP)  /* 按源 IP 地址聚簇 */
7.        r̂_i ← GetRatioBelongToGroup(srcCL, quantile, group_i)  /* 源簇预分组的各组比例值 */
8.        r_i ← GetRatioOfGroup(group_i)  /* 已有分组中各组源簇的比例值 */
9.        d ← max| r_i − r̂_i |  /* 计算偏差值 */
10.       if d ≤ threshold then
11.           AddToGroup(srcCL_i, group_i)
12.           return false  /* 若偏差值不超出设定的阀值,直接将源簇加入所属分组中,并返回 false */
13.       else
14.           AddToGroupWithFilter(srcCL_i, group_i)
15.           return true  /* 若偏差值超出设定的阀值,将源簇有过滤地加入所属分组中,并返回 true */
16.   }
```

5 实验

下面对自适应 DDoS 攻击防御系统进行了测试和评估。如上文所述,一个有效的 DDoS 防御机制必须能快速地检测攻击,过滤掉攻击流量,并最大限度地减少对合法用户流量的影响。出于这个目的,用于评估的度量如下:(1)检测时间,攻击开始和攻击检测之间经过的时间;(2)缓解精度,合法用户流量受影响的程度。

实验的合法流量数据使用了新西兰怀卡托大学 WAND 网络研究组在 2013 年 10 月 21 日公开的“ISPDSL II”[17]部分数据,DDoS 攻击流量数据使用了 CAIDA[18]公开的“DDoS 2007”数据。“ISPDSL II”合法流量负载在 11 Mb/s 至 13 Mb/s,原始攻击流量负载在 500 Kb/s 至 600 Kb/s,本文将“DDoS 2007”攻击流量放大到 12 Mb/s 至 14 Mb/s 后与“ISPDSL II”合法流量进行了合成。实验采用的滑动时间窗口大小为 3 600 s,滑动时间距离为 300 s,合法流量数据从第 900 s(第 10 个时间窗口)开始加入了 500 s 的 DDoS 攻击流量数据。

5.1 分组效果

在评估度量之前,先对分组策略的各分组源簇比例偏差进行了统计。该实验使用了 7 个时间窗口的 DDoS 攻击数据,图 5(a)描述了合法流量在各分组源簇比例偏差值,最大源簇比例偏差值低于 0.004。图 5(b)描述了混合 DDoS 攻击流量在各分组源簇比例偏差值,检测阀值设为 0.004,从含有 DDoS 攻击流量(第 10 个时间窗口)的时间段内,G_0、G_3、G_4和 G_6分组的源簇比例偏差值有十分明显的上升趋势,因此本文的分组策略对检测 DDoS 攻击流量的效果很好。

5.2 检测时间

快速检测出流入的 DDoS 攻击流量对于缓解效果至关重要。图 6 表示出了当发生攻击期间的时刻 t 的源簇分组比例与合法源簇分组比例之间的最大偏差 $(\max_i|r_i-\hat{r}_i|)$。当攻击开始时这种差异迅速增加。如果检测得到这个最大值超过允许的范围,那么这个异常会被检测出来。例如:设检测阀值为 0.004,那么检测攻击要经过 6 s 的时间。

5.3 缓解精度

该实验研究 BF 和 AL 如何有效地防止非法数据

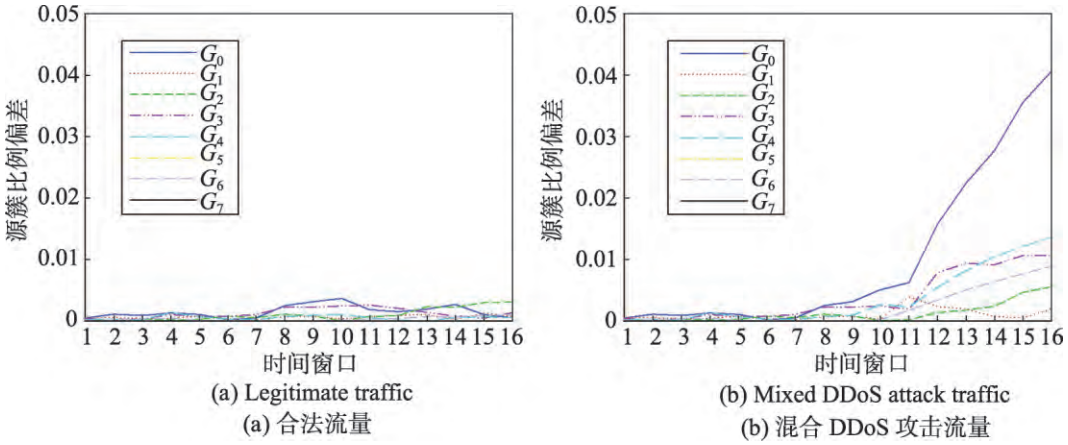


Fig.5　Source cluster proportional deviation of legitimate traffic and mixed DDoS attack traffic in each packet

图5　合法流量及混合DDoS攻击流量在各分组源簇比例偏差值

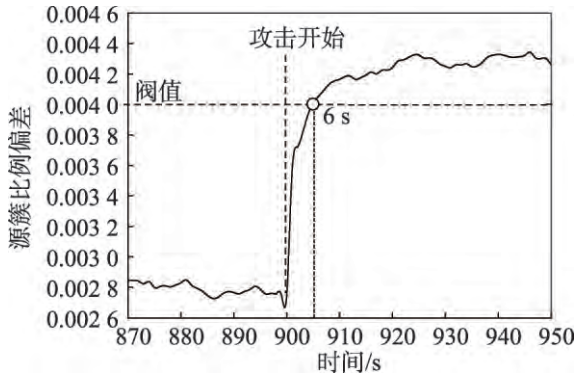


Fig.6　Detection time

图6　检测时间

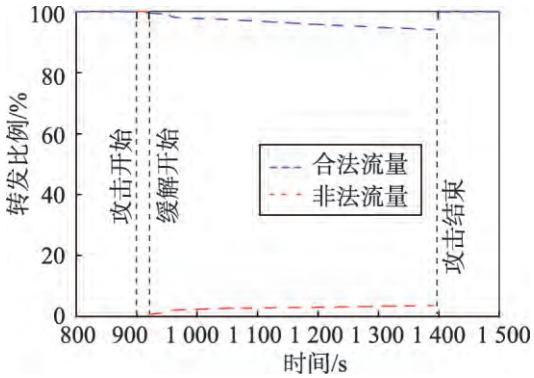


Fig.7　Mitigation precision

图7　缓解精度

转发到受保护的实体上。为了做到这一点,在检测到攻击时(第906 s),系统丢弃所有不属于BF或AL的数据包。首先测量与该MC组件转发的合法流量,并丢弃非法流量的精度。图7给出了合法流量与非法流量被转发到受保护网络实体的比例。可以注意到,在攻击期间,合法流量被转发的百分比约为95%。非法流量在全部转发到受保护实体并实施攻击前被检测,一旦缓解被激活,约超过97%的非法流量会被丢弃。

6　结束语

本文提出了基于Spark Streaming的自适应DDoS攻击防御框架,利用数据流实现应用的实时性要求。通过连续提取源簇的特征和保持分组的源簇数据检测DDoS攻击,对于合法用户来说缓解了DDoS攻击的影响。本文的缓解机制对瞬间拥塞问题也能起到很好的缓解作用,因为用户的优先级与其历史访问次数正相关,若优先级较高则优先转发。该系统已在弹性可扩展的Spark Streaming框架上进行了实现,实验评估显示,它在检测和缓解方面都有非常好的效果。

References:

[1] Bhuyan M H, Bhattacharyya D K, Kalita J K. Network anomaly detection methods, systems and tools[J]. IEEE Communications Surveys & Tutorials, 2014, 16(1): 303-336.

[2] Dean J, Ghemawat S. Mapreduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51 (1): 107-113.

[3] Gulisano V, Jimenez-Peris R, Patino-Martnez M, et al. StreamCloud: a large scale data streaming system[C]//Pro-

ceedings of the 2010 IEEE 30th International Conference on Distributed Computing Systems, Genova, Italy, Jun 21-25, 2010. Piscataway, USA: IEEE, 2010: 126-137.

[4] Kompella R R, Singh S, Varghese G. On scalable attack detection in the network[C]//Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement, Sicily, Italy, 2004. New York, USA: ACM, 2004: 187-200.

[5] Roesch M. Snort[EB/OL]. (2012)[2015-07-10]. http://www.snort.org/.

[6] Lakhina A, Crovella M, Diot C. Mining anomalies using trace feature distributions[C]//Proceedings of the 2005 ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Philadelphia, USA, Aug 22-26, 2005. New York, USA: ACM, 2005: 217-228.

[7] Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey[J]. ACM Computing Surveys, 2009, 41(3): 15.

[8] Silveira F, Diot C, Taft N, et al. Astute: detecting a different class of trace anomalies[J]. ACM SIGCOMM Computer Communication Review, 2010, 40(4): 267-278.

[9] Tang Chenghua, Liu Pengcheng, Tang Shensheng, et al. Anomaly intrusion behavior detection based on fuzzy clustering and features selection[J]. Journal of Computer Research and Development, 2015, 52(3): 718-728.

[10] Cheng Jieren, Yin Jianping, Liu Yun, et al. Detecting distributed denial of service attack based on address correlation value[J]. Journal of Computer Research and Development, 2009, 46(8): 1334-1340.

[11] Liu Yun, Cai Zhiping, Zhong Ping, et al. Detection approach of DDoS attacks based on conditional random fields[J]. Journal of Software, 2011, 22(8): 1897-1910.

[12] Krishnamurthy B, Sen S, Zhang Yin, et al. Sketch-based change detection: methods, evaluation, and applications[C]//Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement, Miami Beach, USA, Oct 27-29, 2003. New York, USA: ACM, 2003: 234-247.

[13] Barford P, Kline J, Plonka D, et al. A signal analysis of network trace anomalies[C]//Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment, Marseille, France, Nov 6-8, 2002. New York, USA: ACM, 2002: 71-82.

[14] Roughan M, Griffin T, Mao Z M, et al. Combining routing and trace data for detection of IP forwarding anomalies[J]. ACM SIGMETRICS Performance Evaluation Review, 2004, 32(1): 416-417.

[15] Cai Zhiping, Wang Zhijun, Zheng Kai, et al. A distributed TCAM coprocessor architecture for integrated longest prefix matching, policy filtering, and content filtering[J]. IEEE Transactions on Computers, 2013, 62(3): 417-427.

[16] Broder A Z, Mitzenmacher M. Network applications of Bloom filters: a survey[J]. Internet Mathematics, 2003, 1(4): 485-509.

[17] WITS: Waikato Internet traffic storage, WAND network reseach group "ISPDSL II" [EB/OL]. (2013)[2015-07-10]. http://wand.net.nz/wits/.

[18] Hick P, Aben E, Clay K. The CAIDA "DDoS Attack 2007" dataset[EB/OL].(2012)[2015-07-10]. http://www.caida.org.

附中文参考文献：

[9] 唐成华, 刘鹏程, 汤申生, 等. 基于特征选择的模糊聚类异常入侵行为检测[J]. 计算机研究与发展, 2015, 52(3): 718-728.

[10] 程杰仁, 殷建平, 刘运, 等. 基于地址相关度的分布式拒绝服务攻击检测方法[J]. 计算机研究与发展, 2009, 46(8): 1334-1340.

[11] 刘运, 蔡志平, 钟平, 等. 一种基于条件随机场的DDoS攻击检测方法[J]. 软件学报, 2011, 22(8): 1897-1910.

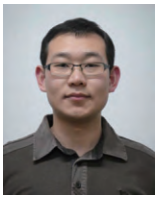
FANG Feng was born in 1987. He is an M.S. candidate at National University of Defense Technology. His research interests include network security and big data processing, etc.

方峰(1987—),男,安徽蒙城人,国防科技大学硕士研究生,主要研究领域为网络安全,大数据处理等。

CAI Zhiping was born in 1975. He received the Ph.D. degree from National University of Defense Technology in 2005. Now he is a professor and M.S. supervisor at National University of Defense Technology. His research interests include network virtualization, network security, big data processing and software-defined networking, etc.
蔡志平(1975—),男,湖南益阳人,2005年于国防科技大学获得博士学位,现为国防科技大学副教授、硕士生导师,主要研究领域为网络虚拟化,网络安全,大数据处理,软件定义网络等。

ZHAO Qijia was born in 1985. He is an M.S. candidate at National University of Defense Technology. His research interests include network security and machine learning, etc.
肇启佳(1985—),男,吉林长春人,国防科技大学硕士研究生,主要研究领域为网络安全,机器学习等。

LIN Jiarun was born in 1987. He is a Ph.D. candidate at National University of Defense Technology. His research interests include cloud computing security, network security and machine learning, etc.
林加润(1987—),男,福建莆田人,国防科技大学博士研究生,主要研究领域为云计算安全,网络安全,机器学习等。



ZHU Ming was born in 1986. He is a Ph.D. candidate at National University of Defense Technology. His research interests include wireless networks, network virtualization and software-defined networking, etc.
朱明(1986—),男,吉林长春人,国防科技大学博士研究生,主要研究领域为无线网络,网络虚拟化,软件定义网络等。