doi: 10.14132/j.cnki.1673-5439.2015.01.013

SDN 环境下基于 KNN 的 DDoS 攻击检测方法

肖 甫[1,2,3],马俊青[1],黄洵松[1],王汝传[1,2,3]

(1. 南京邮电大学 计算机学院,江苏 南京 210023
2. 南京邮电大学 江苏省无线传感网高技术研究重点实验室,江苏 南京 210003
3. 南京邮电大学 宽带无线通信与传感网技术教育部重点实验室,江苏 南京 210003)

摘要: DDoS 攻击是当前互联网面临的主要威胁之一,如何快速准确地检测 DDoS 攻击是网络安全领域研究的热点问题。文中提出了一种在 SDN 环境下基于 KNN 算法的模块化 DDoS 攻击检测方法,该方法选取 SDN 网络的 5 个关键流量特征,采用优化的 KNN 算法对选取的流量特征进行流量异常检测,最后基于 NOX 控制器和 NetFPGA 交换机进行了实验验证。实验结果表明:相对其他的分类检测算法,所提的检测方案具有更高的识别率和更低的误报率。

关键词: 分布式拒绝服务攻击;软件定义网络;K 近邻;OpenFlow
中图分类号: TP393.08;TN915.08 **文献标志码:** A **文章编号:** 1673-5439(2015)01-0084-05

DDoS attack detection based on KNN in software defined networks

XIAO Fu[1,2,3], MA Junqing[1], HUANG Xunsong[1], WANG Ruchuan[1,2,3]

(1. School of Computer Science & Technology, Nanjing University of Posts and Telecommunications, Nanjing 210023, China
2. Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
3. Key Lab of Broadband Wireless Communication and Sensor Network Technology, Ministry of Education, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: DDoS attack is a major threat to current Internet. It is difficult to detect the DDoS attack quickly and accurately. This paper proposes a DDoS detection algorithm using improved K-nearest neighbor (KNN) algorithm in software defined network (SDN) architecture based on traffic flow features, and the feasibility of the method is validated through an experiment based on NOX and NetFPGA. Experimental results demonstrate that the method can obtain a lower false alarm and higher detection result than other classification algorithms.

Key words: DDoS attack; software defined networks (SDN); K-nearest neighbor; OpenFlow

分布式拒绝服务(Distributed Denial of Service, DDoS)攻击是当前互联网面临的主要威胁之一。DDoS 攻击发起者首先利用互联网用户的漏洞,收集大量的傀儡机,然后协同调度这些傀儡机同时伪造数据,发送非法请求导致目标主机瘫痪。DDoS 攻击主要分为两种类型:带宽消耗型和资源消耗型,带宽消耗型攻击通过发送大量无用的数据包给受害主机或网络,占用网络带宽,从而使正常请求流量不可达;资源消耗型攻击通过消耗目标主机的资源,比如 CPU、内存、硬盘等,使目标主机无法回应正常用户的请求。

现有 DDoS 攻击检测方法主要针对传统网络架构,软件定义网络(Software Defined Networks, SDN)作为一种当前出现的新型网络架构,将网络的数据

收稿日期: 2014-09-23 本刊网址: http://nyzr.njupt.edu.cn
基金项目: 国家自然科学基金(61373137,61373017)、江苏省六大人才高峰项目(2013-DZXX-014)和江苏省高校自然科学研究重大项目(14KJA520002)资助项目
通讯作者: 肖 甫 电话: 025-83492410 E-mail: xiaof@njupt.edu.cn

转发层面与网络控制层面相分离,网络数据层面由通用的硬件来实现,而网络的控制层面由独立的软件来实现。OpenFlow 技术作为 SDN 架构的一种实现方式被应用在很多方面,如负载均衡、流量管理、路由,本文将其功能延伸到 DDoS 攻击检测,针对 SDN 架构,提出了一种基于 K 近邻(K-Nearest Neighbor)算法的 DDoS 检测方法。

1　相关工作

目前针对传统网络架构,研究者提出了大量的 DDoS 攻击检测方法,而 OpenFlow[1-2] 技术主要是基于流进行数据转发的,目前已有少量基于流的攻击检测方法。Cheng 等人[3] 提出了基于 IP 流交互的 DDoS 攻击检测方法,该方法使用网络交互特征算法(FIF)构建一个基于 FIF 时间序列的 DDoS 攻击检测模型,能降低背景流干扰,具有更低的误报率和漏报率,但是该方法只使用了网络流的交互特征,判定的全面性不够。Kim 等人[4] 提出了一种基于流信息阈值来判别流量正常的方法,该方法采用思科公司的 NetFlow 技术来进行流特征提取,通过流特征和设定阈值组成的检测函数检测流量,该方法能够检测多种入侵,提高了检测的准确率,但是该方法需要事先预处理收集到的流量,开销较大。

文献[5] 重点探讨了 SDN 环境下的流量工程技术,而对于基于 SDN 的攻击检测研究方面,Mehdi 等[6] 利用 SDN 的可编程性将 4 种入侵检测方法移植到 SDN 环境下,实现了在 SOHO 网络中不损失精度情况下线速的攻击检测。Giotis 等[7] 提出了将 SDN 当前的主要实现方案 OpenFlow 技术和 sFlow 技术结合进行流特征提取,并采用基于熵的算法进行入侵检测。Braga 等[8] 使用 SOM 神经网络方法进行 OpenFlow 流量检测,但是 SOM 收敛速度慢,训练时间长。

针对以上不足,本文提出一种 SDN 环境下的 DDoS 攻击检测方法,该方法通过使用优化的 KNN 算法来处理 SDN 流量的关键特征以判别流量是否正常,并通过在一个以 NOX[9] 为 OpenFlow 控制器,以 NetFPGA[10] 为 OpenFlow 交换机的网络中实现了该算法。实验采用的 NOX 控制器可通过分析流表实现网络关键属性的提取,而以 NetFPGA 为交换机则可方便地将本文方法部署到真实网络环境中进行测试和验证。

2　背景知识

2.1　SDN 介绍

软件定义网络(SDN)作为一种新型的网络架构,将网络的数据转发层面与网络控制层面相分离,网络数据层面由通用的硬件来实现,而网络的控制层面由独立的软件来实现,网络设备与控制软件之间的通信也可以由程序实现。OpenFlow 技术作为 SDN 的一种实现方式,可以方便的管理和配置 SDN。OpenFlow 技术主要由 OpenFlow 交换机和 OpenFlow 控制器两部分组成,OpenFlow 交换机作为数据转发层面主要负责数据流量的转发,OpenFlow 控制器作为控制层面主要负责交换机上流表的配置。控制器与交换机之间可以通过 OpenFlow 协议实现流表的查询、添加、删除等操作,也可以查看交换机各个端口的统计信息。

如图 1 所示,OpenFlow 采用流转发取代了传统的包转发,流是在同一时间间隔内具有相同特征的包的集合,流量进入 SDN 交换机时首先查看交换机上的流表,有匹配项执行相应的行动,比如转发操作;如果没有匹配的表项,则将报文发送给控制器 NOX,由控制器 NOX 决定如何生成流表并发送给交换机。

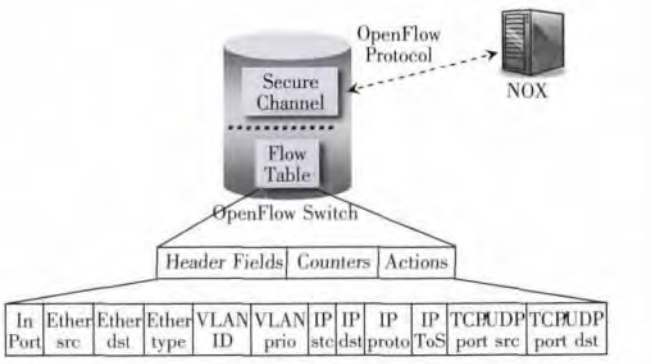

图 1　OpenFlow 结构

2.2　KNN 算法

K 近邻算法(K-Nearest Neighbor Algorithm)作为一种经典的模式识别分类算法,相对其他分类算法而言具有实现简单且支持增量学习的优点,其基本理论可概述为:给定一组已正确分类的数据集作为训练数据集,计算新输入的实例与训练数据集最接近的 k 个实例,在此基础上查看这 k 个实例中的类别数量,将数量最多的类分配给新输入的实例。

KNN 算法的具体步骤为:

设训练集为 training_set,测试集为 test_set,训练集中样本表示为 $n+1$ 维向量 $(x_1, x_2, \cdots, x_n, y)$,测试集中样本表示为 $(x'_1, x'_2, \cdots, x'_n, y')$,其中 x_i 为训练集中样本各维属性值,而 y 为类型值。

算法流程如下:

(1) 准备训练数据并训练;

(2) 存储训练数据结果;

(3) for $(x'_1, x'_2, \cdots, x'_n, y')$ in test_set,计算与 training_set 中样本的欧式距离 dist,

$$\text{dist} = \sqrt{\sum_{i=1}^{n} (x'_i - x_i)^2}$$

(4) 选择与测试样本$(x'_1, x'_2, \cdots, x'_n, y')$距离最近的 k 个样本点,返回数量最多的类别号。

3 基于 KNN 的 DDoS 攻击检测方法

如图 2 所示,本攻击检测方法包括 3 个模块,分别为流表收集模块、流特征提取模块、KNN 分类器模块。流表收集模块定期向 OpenFlow 交换机发送流表请求,交换机回复的流表信息通过加密信道传送给流表收集节点。流特征提取模块负责接收流表收集模块收集的流表,提取出与 DDoS 相关的的 5 个特征组成五元组,每个五元组都以收集到此数据的交换机 ID 来作为标识,从而可以监测哪个交换机发现了 DDoS 攻击。KNN 分类器模块负责将收集到的五元组进行分类,以区分该段时间内流量是 DDoS 攻击流量还是正常流量。

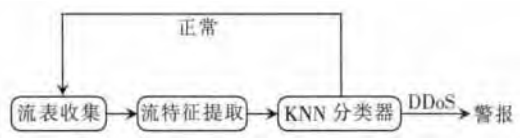

图 2 检测流程图

3.1 流表收集模块

流表收集模块主要通过 OpenFlow 协议来实现流表的收集,交换机回复控制器定期发送 ofp_flow_stats_request 报文,获得的流表将作为流特征提取的输入,获取流表的时间间隔必须适中,因为间隔时间太大会导致在发现 DDoS 攻击前就导致网络瘫痪,而时间间隔太短则会使控制器过载。我们设置流表周期获取时间为 5 秒,其与 NOX 控制器设置的近期未命中流删除时间保持一致。

3.2 流特征选取模块

流特征选取模块我们参考使用文献[11]中的 3 个参数 MPf、PCf、PGS,并引入了另 2 个参数 MBf,SGS,具体为:

(1) 流包数中位值(Median of Packets per flow,MPf)

取流量表中每道流的数据包数作为样本,取样本中位数作为特征向量第一维参数,对应为:

$$MPf = \frac{x_{\lfloor (n-1)/2 \rfloor} + x_{\lfloor n/2 \rfloor}}{2}$$

其中,x_i 表示由每道流数据包数组成的样本集 $X = \{x_1, x_2, \cdots, x_n \mid \forall i < j, x_i \leqslant x_j\}$ 中的第 i 项,n 为样本数。

(2) 流字节数中位值(Median of Bytes per flow,MBf)

取流量表中每道流的字节数作为样本,取样本中位数作为特征向量的第二维参数,对应为:

$$MBf = \frac{y_{\lfloor (n-1)/2 \rfloor} + y_{\lfloor n/2 \rfloor}}{2}$$

其中,y_i 表示由每道流字节数组成的样本集 $Y = \{y_1, y_2, \cdots, y_n \mid \forall i < j, y_i \leqslant y_j\}$ 中的第 i 项,n 为样本数。

(3) 对流比(Percentage of Correlative flow,PCf)

网络流量中正常流量 IP 地址具有交互性,这是因为正常流的目的是为了获取或者提供服务。我们定义流 X 为 FlowX = (srcIP = A,dstIP = B,Protocol = C),流 Y 为 FlowY = (srcIP = B,dstIP = A,Protocol = C),我们称流 X 与 Y 为对流,对流比 PCf 计算方法如下:

$$\text{PCf} = 2 \times \text{Pair\_flow\_num} / \text{flow\_num}$$

其中,Pair_flow_num 是交互流的对数,flow_num 是流的总数。

(4) 端口增速(Ports Generating Speed,PGS)

网络处在在正常情况下时,流量中端口的增加或减少的数目较少,增速稳定在一个区间内,DDoS 攻击不仅使用 IP 欺骗,还会随机生成端口号,所以端口的增速也会显著增大,对应为:

$$\text{PGS} = \text{Port\_num} / \text{interval}$$

(5) 源 IP 增速(Source IP Growing Speed,SGS)

DDoS 攻击发生时,攻击者通常使用 IP 地址欺骗,流的源 IP 地址是由攻击者随机生成的,目的地址一般是受害主机的 IP 地址,所以源 IP 地址的增速会显著增大,所以我们选择源 IP 地址的增速作为属性之一,计算方法如下:

$$\text{SGS} = \text{srcIP\_num} / \text{interval}$$

其中,srcIP_num 为源 IP 地址的数量,interval 为时间间隔。

3.3 KNN 流量识别模块

流量识别模块,接收流特征提取模块传递过来的五元组从而识别流量是否正常。本文使用基于 KNN 算法的分类方法,KNN 算法实现方便,并且支持增量学习,但是 KNN 算法也存在一些缺点,比如计算时间过长,计算开销大。为此我们使用了基于索引的数据结构 KD-Tree 来存放学习样本。KNN 算法在构建学习数据时当学习样本不平衡情况,会导致分类出现很大的误差。因此,我们选择学习样

本时,正常流组成的五元组和 DDoS 攻击流组成的五元组比率采用 1 ∶1。

流量识别对应的伪代码如下:

```
struct Node{
    vec X{ x0 ,x1 ,…} ; //坐标向量
    integer split; //分割域
    integer classify_no; //分类号
    Node* left_child,* right_child; //指向左右孩子
}
@ param root: 当前树的树根
@ param K [ ]: 保存所有用于建树的案例点
Procedure Build( Node root,Node K [MAXN] ,int left,int right)
{
if left < right them:
    return
else:
    sort( K + left,K + right,split) //split 为当前层分割域
    int mid = ( left + right) /2
    K [mid] split←split
    root←K [mid]
    Bulid( K [mid] ,K,left,mid - 1)
    Build( K [mid] ,K,mid + 1,right)
}
//查询 k 个最近邻
@ param a: 当前需要分类的样本点
@ param ans [ ]: 保存最近的 k 个邻居
Procedure Query( Node now,Node & a,Node ans [ ],int k,intp) {
    Node imgine_node    //构造线段优化中的假想点
    if dist( imgine_node,a) > ans [k] . dist then:
    return
else:
    remin( now,a,ans,k)    //使用 ans 数组存放节点 a 的 k 个最近邻
    if( a,vec [p] > now. vec [p] ) {
    query( now. right,a,ans,k,( p + 1) % n)
    query( now. left,a,ans,k,( p + 1) % n)
  }
    else{
    query( now. left,a,ans,k,( p + 1) % n)
    query( now. right,a,ans,k,( p + 1) % n)
  }
}
```

通过在建树时统计子树部分参数,可以对 KD 树的搜索和回溯过程进一步优化,使得搜索更少的点即可获得 K 近邻。建树时,在每个树节点维护以该节点为根的树中每一维的最大值和最小值。在搜索或回溯过程中,在当前点构造距离待求点最近的假想点 q,计算 q 关于待求点的距离,若该距离不小于 dist,则直接放弃该树,设 $X_p = (x_{p1} ,x_{p2} ,\cdots ,x_{pn})$,所采用的剪枝优化示意图如图 3 所示。

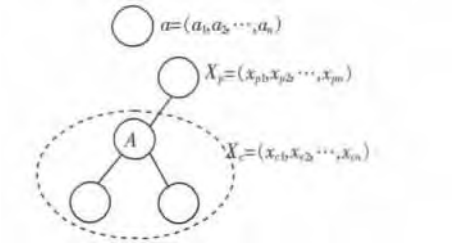

图 3　KD 树剪枝优化

其中,$a = (a_1 ,a_2 ,\cdots ,a_n)$ 是当前需要求 K 近邻的节点,X_c 是以当前检查到的节点 A 为根的树节点的集合,X_c 维护着两个数组 max [ ] = max{ x_{ci}} 和 min [ ] = min{ x_{ci}} ,设 t [ ] = max [ ] ‖ min [ ],为提升算法的执行效率,采用子树剪枝算法,当下式满足时,则对以 A 为根节点的子树进行剪枝操作。

$$\min(dist((a_1 ,a_2 ,\cdots ,a_n) - (t_1 ,t_2 ,\cdots ,t_n))) < dist((a_1 ,a_2 ,\cdots ,a_n) - (x_{p1} - x_{p2} ,\cdots ,x_{pn}))$$

K 近邻算法中节点相似度的计算需要使用距离度量来进行判定,本文先对数据进行归一代处理,再使用欧氏距离计算两点之间的距离。而对于 K 值的选择我们采用交叉验证的方法来确定,最后选择的 K 值为 $\sqrt{m}/2$,其中 m 为训练样本数量。

4　实验及分析

为了验证本文的提出的检测算法是否正确,我们设计了如图 4 所示的实验环境,其中 Network1 是报文发送网络,Gateway 及 Network2 是受害者目标,Network1 可以发送正常报文以产生训练样本中的正常样本,也可以发送 DDoS 攻击报文产生训练样本中的 DDoS 攻击样本,还可以作为测试检测算法时的正常流量和 DDoS 混合流量的发送端。

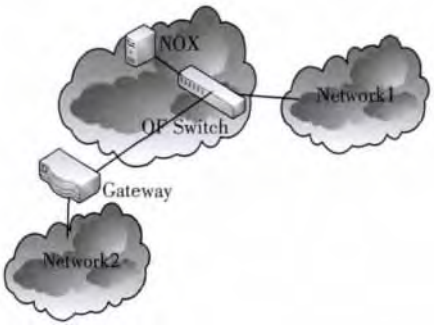

图 4　实验拓扑图

训练样本生成阶段中,正常流量由 Network1 中主机进行正常的网络访问而产生,经统计其中大约有 65% 的 TCP 流量,20% 的 UDP 流量,5% 的 ICMP 流量以及 10% 的其它流量。而对于异常流量的产生我们选用了经典的 DDoS 攻击工具 TFN 产生,对应发起 TCP SYN flood,UDP flood,ICMP flood 等典型

网络流量攻击。在对上述流量通过 NetFPGA 交换机时产生的流表进行收集,产生学习样本,分为两组:正常流量训练样本以及 DDoS 攻击流量训练样本。因为 KNN 检测算法要求训练样本集的类型分布尽量可能平均,在本文中,选择用于训练的 DDoS 流量样本与正常流量样本数目相同。

流量检测阶段中,我们在 NOX 控制器所在主机上每隔 5 秒收集一次 NetFPGA 交换机上的流表,在收集到的流表项中每 50 个流表项提取出一个 5 元组使用 K 近邻方法进行分类,若分类为正常流量,我们认为该 SDN 交换机所在软件定义网络未受到 DDoS 攻击,若分类为非正常流量,则认为该网络遭到了 DDoS 攻击。本文使用两个参数 DR(攻击检出率)和 FA(误警率)来评价本文基于 K 近邻方法的检测算法的效果,同时使用支持向量机(Support Vector Machine,SVM)检测方法作为对照组,支持向量机我们使用在 MATLAB 下安装 libsvm 组件来实现,其中 libsvm 中核函数选用 RBF 函数,对照组中结果如表 1 所示。

表 1 检测算法结果

流表条数	训练样本数	测试样本数	SVM/%		KNN/%	
(正常/DDoS)	(正常/DDoS)	(正常/DDoS)	DR	FA	DR	FA
75 000/75 000	1 500/1 500	1 000/1 000	92.3	0.50	96.4	0.30
150 000/150 000	3 000/3 000	1 000/1 000	93.5	0.70	98.0	0.60

分析检测算法实验结果可以看出,使用本文提出的 KNN 算法检测 5 元组比使用 SVM 检测方法有更高的检出率和更低的误警率。而在使用同样分类算法的情况下训练样本量越多,分类的检出率越高,但误警率也同时上升,样本数越高,相应的训练时间以及检测时间变长,对应的时间开销如表 2 所示。

表 2 时间开销

训练样本数	学习时间/ms	检测时间/μs	
		KNN	枚举法
3 000	112	230	1 148
6 000	225	422	2 120

本文 KNN 检测算法时间复杂度为 $O(n)$,其中 n 为训练样本的数量,各次实验的开销如表 2 所示。在优化条件下,KNN 检测算法实际运行效率大约是枚举法求取所有点距离方法的 5 倍。

5 结束语

本文提出了一种面向 SDN 架构的 DDoS 攻击检测方法,该方法基于优化的 KNN 算法处理流量的关键属性从而判定流量是否为 DDoS 流量,该方法的优点在于较为全面地提取和分析 SDN 架构下流量关键属性,相对于其他的分类检测算法有更好的检出率和更低的误警率。

参考文献:

[1] MCKEOWN N, ANDERSON T, BALAKRISHNAN H, et al. OpenFlow: Enabling innovation in campus networks [J]. ACM SIGCOMM Communication Review, 2008, 38(2): 69 – 74.

[2] OpenFlow Consortium. OpenFlow website [EB/OL]. http://archive.OpenFlow.org/.

[3] CHENG J, TANG X, ZHU X. Distributed denial of service attack detection based on IP Flow Interaction [C] // IEEE International Conference on E-Business and E-Government (ICEE). 2011: 1 – 4.

[4] KIM M, KANG H, HUNG S, et al. A flow-based method for abnormal network traffic detection [C] // IEEE Network Operations and Management Symposium. 2004: 599 – 612.

[5] MEHDI S A, KHALID J, KHAYAM S A. Revisiting traffic anomaly detection using Software Defined Networking [C] // Proceedings of the 14th International Conference on Recent Advances in Intrusion Detection. 2011: 161 – 180.

[6] GIOTIS K, ARGYROPOULOS C, ANDROULIDAKIS G. Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments [J]. Computer Networks, 2014, 62(7): 122 – 136.

[7] BRAGA R, MOTA E, PASSITO A. Lightweight DDoS flooding attack detection using Software Defined Networking [C] // The 35th Annual IEEE Conference on Local Computer Networks. 2011: 161 – 180.

[8] AGARWAL S, KODIALAM M, LAKSHMAN T V. Traffic Engineering in Software Defined Networks [C] // IEEE INFOCOM. 2013: 2211 – 2219.

[9] GUDE N, KOPONEN T, PETTIT J, et al. NOX: Towards an operating system for networks [J]. ACM SIGCOMM Communication Review, 2008, 38(3): 105 – 110.

[10] Nicira. NOX repository website [EB/OL]. http: // www.noxrepo.org/.

[11] GUO R, YIN H, WANG D, et al. Research on the active DDoS filtering algorithm based on IP flow [C] // IEEE 5th International Conference on Natural Computation. 2009: 628 – 632.

作者简介:

肖 甫(1980 –),男,湖南邵阳人。南京邮电大学计算机学院教授,博士生导师。目前主要从事无线传感器网络、计算机网络等方面的研究。