基于 MapReduce 的僵尸网络在线检测算法

蒋鸿玲*① 邵秀丽① 李耀芳②
①(南开大学信息技术科学学院 天津 300071)
②(天津城市建设大学计算机与信息工程学院 天津 300384)

摘 要：目前僵尸网络主要是通过网络流量分析的方法来进行检测，这往往依赖于僵尸主机的恶意行为，或者需要外部系统提供信息。另外传统的流量分析方法计算量很大，难以满足实时要求。为此该文提出一种基于 MapReduce 的僵尸网络在线检测算法，该算法通过分析网络流量并提取其内在的关联关系检测僵尸网络，并在云计算平台上进行数据分析，使数据获取和数据分析工作同步进行，实现在线检测。实验结果表明该算法的检测率可达到 90%以上，误报率在 5%以下，并且数据量较大时加速比接近线性，验证了云计算技术在僵尸网络检测方面的可行性。

关键词：云计算；僵尸网络；在线检测； MapReduce
中图分类号：TP393 **文献标识码**： A **文章编号**：1009-5896(2013)07-1732-07
DOI: 10.3724/SP.J.1146.2012.01444

Online Botnet Detection Algorithm Using MapReduce

Jiang Hong-ling① Shao Xiu-li① Li Yao-fang②
①(*College of Information Technical Science*, *Nankai University*, *Tianjin* 300071, *China*)
②(*College of Computer and Information*, *Tianjin Institute of Urban Construction*, *Tianjin* 300384, *China*)

Abstract: Most current botnet detection approaches are based on analyzing network traffic and they usually rely on malicious behaviors of bots or need information provided by external systems. Besides, the huge computation of traditional approaches is difficult to meet the real time requirement. So an online botnet detection approach is proposed based on MapReduce. The approach detects botnet by analyzing network traffic and extracting intra relationship of flows. The data analysis is carried out in cloud platform which makes the data capture and data analysis working simultaneously and realizes online detection. The experimental results show that the detection rate of the approach can achieve 90% and the false positive rate is below 5%. When the data is large, the speedup is close to linear. It proves the feasibility of applying cloud computing technologies to botnet detection.

Key words: Cloud computing; Botnet; Online detection; MapReduce

1 引言

僵尸网络(botnet)的快速发展使因特网面临严重的安全威胁。僵尸网络是攻击者(botmaster)通过传播僵尸程序控制大量主机，通过(Command and Control, C&C)命令与控制信道与僵尸主机通信并发布命令[1]。攻击者利用僵尸网络可发起多种攻击，如分布式拒绝服务攻击、垃圾邮件、信息窃取等[2]。

目前僵尸网络检测技术主要是通过分析网络流量检测被感染的主机。文献[3,4]通过识别僵尸主机的恶意行为检测僵尸网络。文献[5]利用 PageRank 计算主机级别再根据已知的僵尸主机信息检测僵尸网络。文献[6]将已知的僵尸主机作为种子节点，根据同一个僵尸网络中的主机至少访问一个相同的僵尸主机的特点，找出与种子节点在同一个僵尸网络中的其他僵尸主机。文献[7]通过网络通信图识别 P2P 网络再利用外部系统提供的信息区分合法 P2P 网络和 P2P 僵尸网络。文献[8]基于昵称检测 IRC (Internet Relay Chat)僵尸网络。

僵尸网络既有恶意攻击行为，也存在命令与控制(C&C)通信，因此僵尸网络既会产生恶意攻击流量，也会产生命令控制流量。现有一些检测方法依赖于恶意攻击流量，但是僵尸主机不会持续不断地进行恶意攻击，在 Botmaster 给僵尸主机发布命令前，僵尸主机保持“发呆”状态，不产生恶意流量[9]，使依赖恶意行为的僵尸网络检测方法失效。此外，僵尸主机的攻击往往很隐蔽，很难通过网络流量检

2012-11-12 收到，2013-02-01 改回
天津市重点项目(11jczdjc28100)和国家科技支撑计划(2012BAF12B00)资助课题
*通信作者：蒋鸿玲 hellojhl@163.com

测出来[10]。相比之下，C&C 通信存在于僵尸网络的整个生命周期中，因此本文通过识别 C&C 通信检测僵尸网络。

另一方面，在当前高速网络环境下，现有通过网络流量分析的检测方法通常先获取流量然后再进行流量分析，并且海量数据的分析工作在单个服务器上串行处理，效率很低，导致检测完成后僵尸网络已经发起大规模攻击，因此本文设计了云计算环境下的并行在线检测算法，使数据获取和分析工作同步进行。云计算具有强大的数据处理能力，提供了处理海量数据的并行编程模型 MapReduce，为基于网络流量分析的僵尸网络检测提供了一个高效的解决方案。目前利用云计算检测僵尸网络也有一些研究成果，如文献[11,12]。文献[11]利用 PageRank 算法检测分布式僵尸网络。文献[12]则关注检测恶意行为。

针对现有方法存在的局限性，本文提出一种通过提取网络流量内在的关联关系检测僵尸网络的算法，并设计了基于 MapReduce 的在线检测方法。该方法的特点是：(1)根据 C&C 流量检测僵尸网络。(2)不依赖僵尸主机的恶意行为，在僵尸主机处于空闲状态或者攻击隐蔽的情况下都能检测出僵尸网络。(3)不需要外部系统提供僵尸网络的信息，可独立完成检测。(4)只分析数据包头部的信息，不涉及隐私问题，能检测加密的僵尸网络。(5)利用云计算实现僵尸网络的在线检测。

2　僵尸网络中的流间关联关系

集中式僵尸网络为防止单点失效通常采用多个 C&C 服务器[13]，将一个域名对应多个 C&C 服务器的 IP 地址。僵尸主机频繁地/周期性地通过 PING-PONG 命令与这些服务器保持连接[14]。分布式僵尸网络没有集中的 C&C 服务器，通常采用发布/订阅方式。控制者发布命令或者更新，僵尸主机频繁向其邻居节点搜索命令[15]。此外，僵尸主机频繁连接邻居节点以交换 keep-alive 消息[3]。分布式僵尸网络中每个僵尸主机维护一个邻居节点列表，并频繁访问列表中的节点，反复连接相同的一组节点[16]。上述集中式和分布式僵尸网络模型中，每个僵尸主机维护一个节点列表，频繁访问列表中的节点以获取命令或者更新。大部分僵尸网络符合该模型，本文关注该类僵尸网络。僵尸主机连接其列表中前后相邻节点的流总是“相继出现”，即一个流先出现，另一个流紧跟前一个出现，这些流具有前驱后继的关系，本文称之为“关联关系”。根据关联关系中流的个数不同，分为二级和多级关联关系。二级关联关系指前后两个流的前驱后继关系，多级关联关系指前后多个流的前驱后继关系。相比之下，合法用户的网络行为较随机，合法的流之间不会呈现明显的关联关系。因此，可通过提取流间的关联关系识别 C&C 流量，从而检测出僵尸主机。

3　基于 MapReduce 的僵尸网络在线检测模型与算法

图 1 是基于 MapReduce 的僵尸网络在线检测模型。该模型在 Hadoop 云平台上进行数据分析，并采用时间窗口机制，数据获取模块持续地捕获数据，在捕获下一个时间窗口数据的同时，数据分析模块处理前一个时间窗口的数据。如图 1 所示数据获取模块先捕获数据 D_i，在数据分析模块处理 D_i 的同时，获取数据 D_{i+1}，使数据获取和数据分析同步进行，实现在线检测。

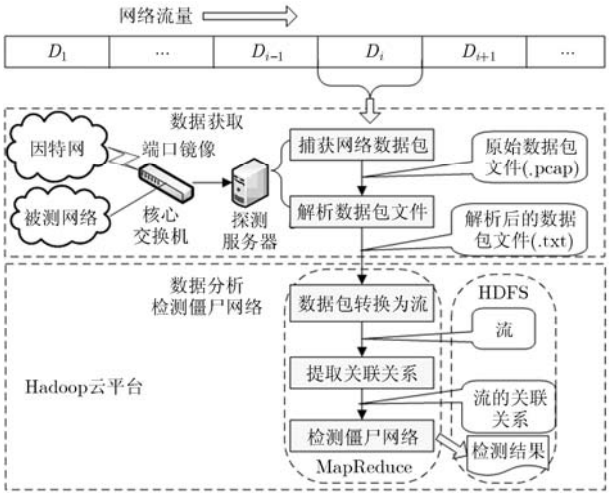


图 1　基于 MapReduce 的僵尸网络在线检测模型

数据获取模块分为两步。第 1 步捕获网络数据包。采用核心交换机端口镜像技术将被测网络的流量镜像到探测服务器，探测服务器运行 tcpdump 工具捕获 TCP 和 UDP 数据包，因为僵尸网络通常采用 TCP 或者 UDP 协议进行通信。第 2 步解析原始数据包文件。将该时间窗口的原始数据包文件转换为可读的文本文件，然后将文件自动上传到 Hadoop 的分布式文件系统 HDFS 中。

数据分析模块检测僵尸网络，采用基于 MapReduce 的僵尸网络在线检测算法，该算法主要分为 3 步。第 1 步形成流，将捕获的原始 TCP/UDP 数据包转换为 TCP/UDP 流。本文中的流指被测网络内的主机和被测网络外的主机间通信产生的一组连续数据包集合。因为僵尸主机间的一次通信会产生多个数据包，分析数据包间的关联关系会产生冗余信息，而且分析数据包的计算量远远大于分析流

的计算量，因此先将数据包转换为流。第 2 步提取流间的关联关系。僵尸主机由于频繁访问同一组节点，连接这些节点的流具有关联关系，而合法用户由于行为随机通常不具有关联关系，因此通过提取关联关系检测僵尸网络。第 3 步检测僵尸网络。根据提取出的关联关系给被测网络内每台主机评分，分数高的是可疑的僵尸主机。

3.1 流的形成

要提取流间的关联关系，首先将原始 IP 数据包转换为流。根据 IP 地址、端口号和协议可以确定一个流，即有相同 5 元组(本地 IP 地址，本地端口号，远程 IP 地址，远程端口号，协议)的数据包集合构成一个流。同时满足相邻两个数据包的时间间隔小于 T_{in}(实验中 T_{in}=60 s)。其中本地 IP 地址和端口号是被测网络内主机的 IP 地址和端口号，远程 IP 地址和端口号是被测网络外的主机的 IP 地址和端口号。

由于僵尸主机间通常采用 TCP 或者 UDP 协议通信，因此只考虑 TCP 和 UDP 流。TCP 流的形成较 UDP 流复杂，本文以 TCP 流为例说明流的形成。首先按照 5 元组对数据包分组，组中按照时间排序，每组从第 1 个数据包开始扫描，找出 TCP3 次握手包作为流的开始，TCP4 次握手包作为流的结束，从开始到结束的所有数据包组成一个 TCP 流。根据该过程设计 MapReduce 并行算法，最后输出每个 TCP 流及其属性。

3.2 关联关系提取

本文根据流的时间信息提取关联关系。由于网络流量巨大，考察每对流是否具有关联关系不现实。为了高效准确地提取出关联关系，本文采取两个措施：第一，只考察在同一个时间窗口内出现次数之差小于阈值 nTh 的流(实验中 nTh=1)。第二，只考察出现时间间隔小于阈值 tTh 的流之间是否具有关联关系(实验中 tTh=60 s)，因为具有关联关系的流通常出现的时间间隔较短。图 2 是关联关系提取过程。分 3 个任务 Job1, Job2 和 Job3 提取二级关联关系。Job4 根据二级关联关系提取多级关联关系。

Job1 计算流出现的次数。根据上述的第 1 个措施，只考察在相同时间段内出现次数之差小于阈值 nTh 的流，因此首先计算当前时间窗口流的出现次数。提取关联关系时只需考虑流的 3 个属性：被测网络内主机的 IP 地址 localIP，与 localIP 通信的远程主机的 IP 地址 remoteIP，流的开始时间 startTime。

Job2 提取候选二级关联关系。先按本地 IP 地址对流进行分组。对每个内部主机 H，形成以 H 为 localIP 的一组流 $G_H=\{f_i\}_{i=1,2,\cdots,m}$。将 G_H 中的流按照开始时间排序，然后依次扫描每一个流，找出在当前流后出现，并且与当前流的时间间隔小于 tTh 的所有后续流。如果这些后续流的出现次数和当前流相差小于 nTh，则这些流和当前流可能具有关联关系。Job2 是提取二级关联关系的核心，其 MapReduce 程序运行示例如图 3 所示。将 Job1 的输出作为 Job2 的输入，包括流的 localIP, remoteIP，开始时间 t 和出现次数 n。MapReduce 将输入数据划分到不同 Map 任务中执行，localIP 相同的记录分到同一个 Reduce 任务中，Reduce 按照上述规则提取候选二级关联关系，输出每个被测主机的候选二级关联关系，将作为 Job3 的输入。

Job3 提取可信的二级关联关系。合法的流偶然也会相继出现，因此要从候选二级关联关系中找出可信的二级关联关系。本文引入置信度 c 来评估关联关系的可信程度。给定候选关联关系(F_i,F_j)，其置信度 c 定义如式(1)所示。如果置信度 c 大于阈值 cTh，则认为该候选关联关系是可信的，c 的取值范围是[0,1]。

$$c(F_i,F_j)=\sqrt{\frac{T_{ij}^2}{N_iN_j}}\times\frac{1}{1+e^{-\min\{N_i,N_j\}}} \tag{1}$$

Job4 提取多级关联关系。合法流由于偶然因素也会有二级关联关系，但合法用户由于行为随机，通常没有多级关联关系，而僵尸主机的流不仅有二级关联关系也有多级关联关系，因此在二级关联关系的基础上提取多级关联关系来检测僵尸主机。多级关联提取是不断迭代的过程，k+1 级关联关系通过 k 级和二级关联关系提取。如存在二级关联关系(F_1,F_2)和(F_2,F_3)可推出三级关联关系(F_1,F_2,F_3)，如果(F_1,F_2,F_3)是可信的，又存在可信的二级关联关系(F_3,F_4)，则可推出四级关联关系(F_1,F_2,F_3,F_4)。给定 k 级关联关系(F_1,F_2,$\cdots$,F_k)，其置信度如式(2)所示，置信度大于阈值 cTh 的认为是可信的。

$$c(F_1,F_2,\cdots,F_k)=\prod_{i=1}^{k-1}c(F_i,F_{i+1}) \tag{2}$$

3.3 僵尸网络检测

提取出关联关系后，根据关联关系的置信度给被测网络内的每台主机评分，分数高的是可疑的僵尸主机。先根据式(3)计算被测网络内每台主机各级关联关系的分数，其中 k 表示 k 级关联关系，n 表示主机 H 的 k 级关联关系个数，c_i^k 是第 i 个 k 级关联关系的置信度，这里只考虑可信的关联关系。

$$S_H^k=\frac{1}{n}\sum_{i=1}^{n}c_i^k \tag{3}$$

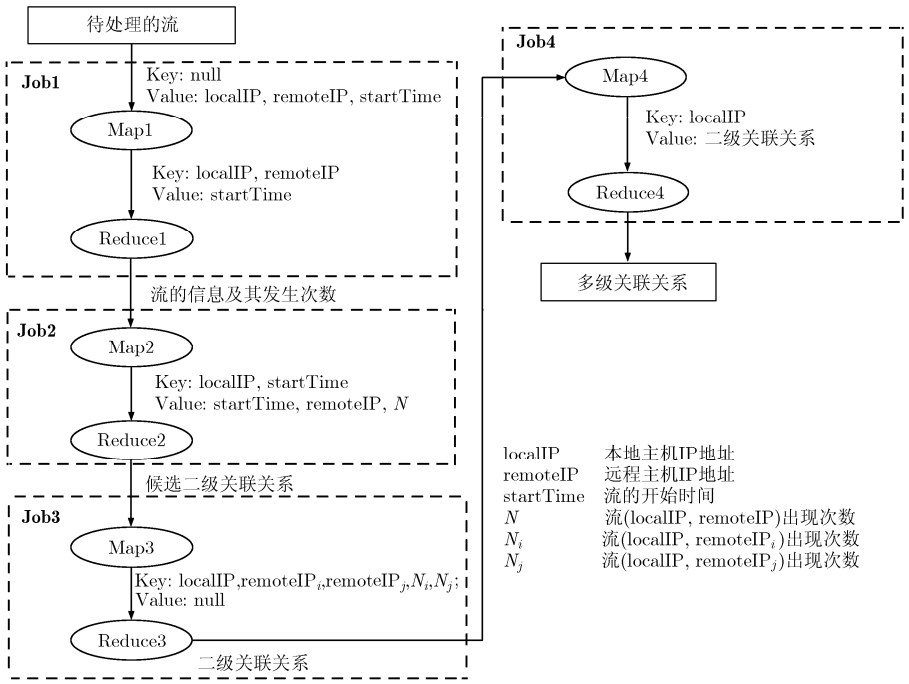


图 2 关联关系提取过程

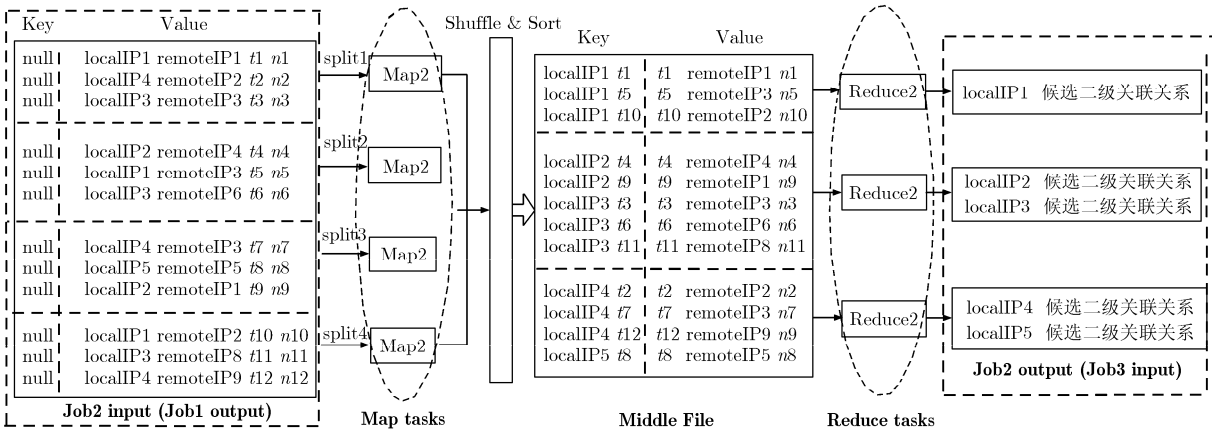


图 3 Job2 的 MapReduce 程序运行示例

再通过式(4)计算每台被测主机的分数。其中 m 为最大关联关系级别，m 越大所考察的关联关系级别越大，越能区分出合法主机和僵尸主机。w_k 为第 k 级关联关系的权值，由于合法主机通常没有多级关联关系，而僵尸主机的关联关系级别较高，因此，随着关联关系级别 k 增大，w_k 也增大，以便更好地区分出正常主机和僵尸主机。

$$S_H = \sum_{k=2}^{m} w_k S_H^k, \quad w_k = \frac{k}{\sum_{k=2}^{m} k} \tag{4}$$

4 实验与结果分析

4.1 实验环境与数据集

本文的实验环境配置如下：10 台双核计算机，其中 1 个 Master 节点，9 个 Slave 节点。基于 Oracle VM VirtualBox4.1.12 的虚拟化环境，Hadoop 0.20.2 搭建的云平台。每台虚拟机安装 Ubuntu 10.10 系统。

实验所用数据集一部分是 MIT 林肯实验室公开的 DARPA 数据集，该数据集由美国国防部资助，由 MIT 林肯实验室在仿真局域网中采集，是最具权威的数据集之一，也是评估入侵检测系统的标准数据集[17]。本文使用的是 1999 年采集的第 1 周无攻击的 tcpdump 数据集，记为 $D1$。虽然 $D1$ 的流量相对较小，但 $D1$ 包含了各种应用的网络流量，流量较全面并且与真实环境基本一致。用 $D1$ 能够检验本文方法在数据量较小时的性能，同时可与第 2 部分数据集比较，检验本文方法在不同网络环境下的效果。第 2 部分数据集为某大学校园网上采集的真实

流量。被测子网内主机约 200 台，白天的网络流量为 150 Mbps~200 Mbps。实验中采用了 2011 年 4 月 6 日的数据，按不同时段分为 3 个数据集 $D2$, $D3$, $D4$。由于具有相同协议的流才具有关联关系，本文实验只分析了 TCP 协议，UDP 协议的分析过程类似。数据集的统计信息如表 1。

表 1 数据集统计信息

数据集	持续时间	TCP 数据包个数	文件大小 (GB)
$D1$	5 d	6757655	0.5
$D2$	3 h (8:00–11:00)	23631638	1.6
$D3$	2 h (11:00–13:00)	11570835	0.8
$D4$	11 h (13:00–24:00)	41309789	3.0

真实的僵尸网络流量数据难以获取，为了验证本文检测算法的有效性，受文献[5,7]的启发，实验中的僵尸网络数据模拟产生。模拟流量依据的僵尸网络模型如第 2 节所述，即每个僵尸主机维护一个节点列表，并频繁地访问列表中的节点以获取命令或者更新。该僵尸网络模型根据大量现有工作总结出来，大多数僵尸网络符合该模型，因此实验中模拟该类僵尸网络的流量。分别以 $D1$~$D4$ 作为背景流量，加入模拟的僵尸网络流，然后随机选取背景流量中若干台主机，替换僵尸主机的 IP 地址为这些主机的 IP 地址，使这些主机既有正常流量也有僵尸网络流量，更符合真实场景。

4.2 实验结果

本节主要从僵尸网络检测的准确性和效率两个方面对检测算法性能进行评价。准确性用检测率(Detection Rate, DR)和误报率(False Positive Rate, FPR)两个指标评估。DR 为僵尸主机被正确检测出来的概率，FPR 为合法主机被错误地检测为僵尸主机的概率。

实验中分别在数据集 $D1$~$D4$ 中注入模拟的僵尸网络流量。$D1$ 中有 5 个僵尸主机从开始到结束一直存在，有 10 个是等待一个随机时间后开始发作的；$D2$ 和 $D3$ 分别有 10 个是一直存在的，有 20 个是随机时间后开始发作的；$D4$ 有 20 个是一直存在的，20 个是随机时间后开始发作的。图 4 是最大关联关系级别 m 取不同值时算法的检测率和误报率。

从图 4(a)可见，m 越大 DR 越低，因为一些僵尸主机的关联关系级别小于 m。数据集 $D1$~$D4$ 的 DR 都大于 0.8，说明本文算法有较高的检测率。从图 4(b)可以看出 m 为 2 时，FPR 较高，因为合法的流偶然也会有二级的关联关系。当 m 增大到 FPR 都小于 0.15, m 越大 FPR 越低，m 大于 6 时 FPR 都为 0，因为合法的流通常没有多级关联关系，验证了通过关联关系区分合法主机和僵尸主机的可行性。从图 4 中可以看出，随着 m 值的增大 DR 降低的同时 FPR 也降低了，因此要选择合适的 m 值既能使检测率在可接受范围又能降低误报率。从图 4 中可以看出 m 取 4 到 6 时，检测率都在 0.9 以上，而误报率都小于 0.05。

图 5 是随时间窗口个数增加算法的检测率，分别以数据集 $D1$ 和 $D3$ 作为背景流量，僵尸网络流量如上所述。图 5(a)是数据集 $D1$ 的检测率，实验中将时间窗口大小设置为 10 h，比较了前 11 个时间窗口的检测率。从图 5(a)可以看出，在第 1 个时间窗口由于僵尸主机的流量较小，还未呈现明显的关联关系，检测率较低，从第 2 个时间窗口开始，僵尸主机流间的关联关系逐渐呈现出来，检测率都在 0.8 以上。图 5(b)为数据集 $D3$ 的检测率，实验中将时间窗口大小设置为 15 min，比较了 8 个时间窗口的检测率。与 $D1$ 类似，$D3$ 在第 1 个时间窗口检测率较低，从第 2 个时间窗口检测率都在 0.7 以上。由此可见，本文的在线检测算法能有效检测出新出现的僵尸主机，并具有较高的检测率。

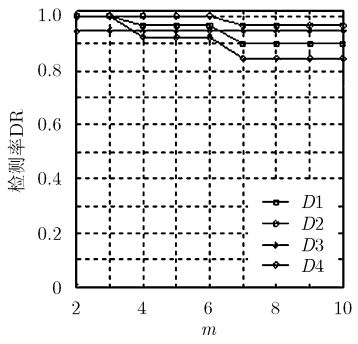
算法的效率用加速比评估。加速比(speedup, su)是指数据集固定，不断增大计算节点个数时算法并行的性能[18]，如式(5)。p 是计算节点个数，T_1 是 1 个节点时的运行时间，T_p 是 p 个节点时的运行时间。

$$su(p) = T_1 / T_p \tag{5}$$

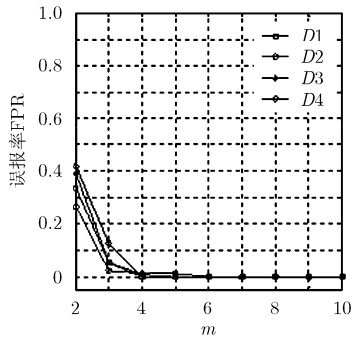
理想情况下，加速比是呈线性的，但实际的加速比要低于理想状态。图 6 是数据集 $D1$~$D4$ 的加速比，T_p 取 3 次实验的平均值。其中图 6(a)是整个检测过程的加速比，可以看出本文提出的基于 MapReduce 的僵尸网络在线检测算法有较好的加速比。$D1$ 的加速比小是因为 $D1$ 的数据量较小，部分节点处于空闲状态。随着数据规模增大，加速比更接近线性，但没有达到线性是因为节点通信、任务启动、调度等开销。图 6(b)是流生成过程的加速比，可以看出图 6(b)的加速比比图 6(a)更接近线性，因为图 6(b)的输入是数据包文件，由于数据包文件较大，多个节点计算效率更高，而关联关系提取的输入是流文件，流文件较小，少量节点即可完成计算，多个节点的效率并不会提升很多，所以整个检测过程的加速比要比流生成过程低一些。该实验同时也证实了本文提出的基于 MapReduce 的僵尸网络在线检测算法更适用于大规模数据集。

4.3 与现有工作比较

现有僵尸网络检测方法很多，所关注的流量数据的种类和特点也不同，目前还没有通用的检测方法。本文通过识别僵尸网络的 C&C 流量来检测僵

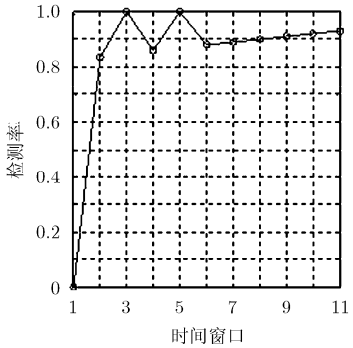


(a) DR

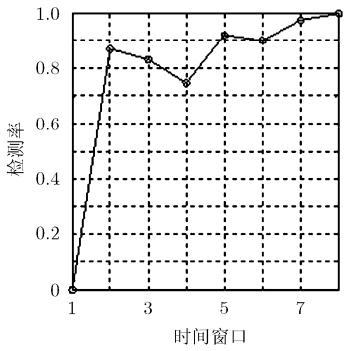


(b) FPR

图 4 不同 m 值算法的检测率和误报率

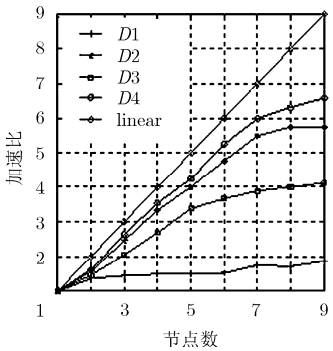


(a) $D1$数据集

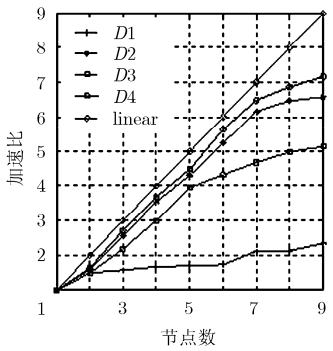


(b) $D3$数据集

图 5 随时间窗口个数增加算法的检测率



(a)整个检测过程的加速比

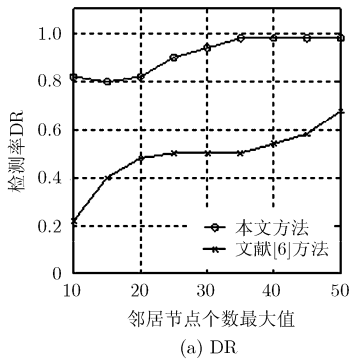


(b)流生成过程的加速比

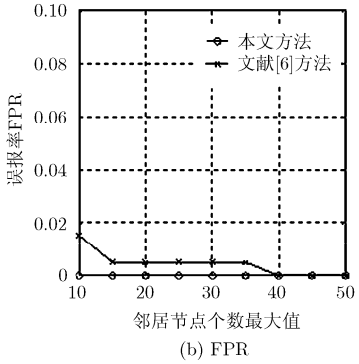
图 6 加速比

尸网络，文献[6]也是通过识别 C&C 流量来检测僵尸网络，并且文献[6]方法是近年来较为经典的僵尸网络检测方法，因此实验中与该方法进行比较。文献[6]方法依据的是同一个僵尸网络中的僵尸主机会至少访问一个相同的节点，将已知的僵尸主机作为种子节点，迭代地找出与种子节点访问相同节点的其他僵尸主机，从而检测出与种子节点在同一个僵尸网络中的僵尸主机。实验中用 $D4$ 作为背景流量，加入模拟的僵尸网络流量，使被测网络中共 50 个僵尸主机，每个僵尸主机的邻居列表中节点个数最小值为 5，最大值为 x, x 从 10 到 50 不断增大。实验用相同的数据集比较了本文方法和文献[6]方法的检测率 DR 和误报率 FPR。图 7 给出了两种方法的比较结果。

实验中本文方法的 m 值取 6。文献[6]中的参数与原文保持一致，即外部节点度的阈值 k=5，平滑系数 β=2。从图 7 中可以看出，随着 x 增大，本文方法和文献[6]方法的检测率都提高，但本文方法的检测率明显高于文献[6]方法。因为当邻居列表中节点较少时，由于邻居节点是随机选择，不同的僵尸主机的访问的相同节点的个数较小，与种子节点隔离的节点较多，所以文献[6]的方法检测率较低。而本文方法不仅仅依赖访问的相同节点个数，更主要的是取决于流间关联关系，因此当邻居节点个数较少时检测率也较高。随着 x 增大，本文方法的误报率都为 0，文献[6]方法的误报率减小。因为僵尸网



(a) DR



(b) FPR

图 7 本文与文献[6]方法的检测率和误报率比较

络中的流间关联关系级别较高，合法主机即使存在关联关系其级别也较低，根据上述实验得知 m 取 6 时，误报率几乎为 0。文献[6]中的方法由于不同的合法节点也有可能访问相同的节点，因此存在误报，但误报率都较低。

与文献[6]方法相比，本文方法有较高的检测率，都在 80%以上，并且误报率为 0。另外，文献[6]的方法需要一个种子节点，即需要依赖外部系统得到已知的僵尸主机，而本文方法不需要事先知道某一个僵尸主机，能独立完成检测。当然，任何性能的提升都要付出一定的代价。本文方法达到了较高的检测率和较低的误报率，但是本文方法的计算量较大。为此本文采用了 MapReduce 并行检测算法，并在 Hadoop 云环境下实现在线检测，使检测效率大大提高，不但取得了良好的检测性能也具有较高的检测效率。

5 结束语

本文讨论了僵尸网络的流间关联关系，并利用云计算的 MapReduce 编程模型，设计了僵尸网络在线检测算法。该方法通过识别 C&C 流的内在关联关系检测僵尸主机。同时在云计算平台上进行在线数据分析，使数据获取与分析工作同步进行，可以在僵尸网络发起大规模攻击前及时发现被感染的主机。实验结果表明，利用流间的关联关系能有效地区分出合法主机和僵尸主机。考察的最大关联关系级别 m 越大，误报率越低，同时检测率也可能降低。设置合理的 m 值才能达到两者的平衡。此外，本文的检测算法对于较大数据流量检测效率更高，在线检测时间越长检测结果越可靠。本文为云计算环境下的海量数据分析与僵尸网络检测奠定了基础，未来的研究工作将进一步检测云环境下的僵尸网络。

参 考 文 献

[1] Lu Wei, Rammidi G, and Ghorbani A. Clustering botnet communication traffic based on n-gram feature selection[J]. *Computer Communications*, 2011, 34(3): 502–514.

[2] 江健, 诸葛建伟, 段海新, 等. 僵尸网络机理与防御技术[J]. 软件学报, 2012, 23(1): 82–96.

[3] Gu Guo-fei, Perdisci R, Zhang Jun-jie, *et al*.. BotMiner: clustering analysis of network traffic for protocol- and structure-independent botnet detection[C]. Proceedings of the 17th Conference on Security Symposium, San Jose, CA, USA, 2008: 378–393.

[4] Gu Guo-fei, Porras P, Yegneswaran V, *et al*.. BotHunter: detecting malware infection through IDSdriven dialog correlation[C]. Proceedings of the 16th USENIX Security Symposium, Boston, MA, USA, 2007: 167–182.

[5] Francois J, Wang S, State R, *et al*.. BotTrack: tracking botnets using NetFlow and PageRank[J]. *Lecture Nodes in Computer Science*, 2011, 6640: 1–14.

[6] Coskun B, Dietrich S, and Memon N. Friends of an enemy: identifying local members of peer-to-peer botnets using mutual contacts[C]. Proceedings of Annual Computer Security Applications Conference, Austin, TX, USA, 2010: 131–140.

[7] Nagaraja S, Mittal P, Hong C, *et al*.. BotGrep: finding P2P bots with structured graph analysis[C]. Proceedings of the 19th USENIX Conference on Security, Washington, USA, 2010, 7: 1–16.

[8] Goebel J and Holz T. Rishi: identify bot contaminated hosts by irc nickname evaluation[C]. Proceedings of USENIX HotBots'07, Berkeley, CA, USA, 2007: 163–174.

[9] Prasad K, Reddy A, and Karthik M. Flooding attacks to internet threat monitors (ITM): modeling and counter measures using botnet and honeypots[J]. *International Journal of Computer Science & Information Technology*, 2011, 3(6): 159–172.

[10] Zhang Jun-jie, Perdisci R, Lee W, *et al*.. Detecting stealthy P2P botnets using statistical traffic fingerprints[C]. Proceedings of IEEE/IFIP 41st International Conference on Dependable Systems and Networks, Hong Kong, China, 2011: 121–132.

[11] Jerome F, Shaonan W, Walter B, *et al*.. BotCloud: detecting botnets using MapReduce[C]. International Workshop on Information Forensics and Security - WIFS, Foz do Iguaçu, Brazil, 2011: 1–6.

[12] Zhao Yao, Xie Ying-lian, Yu Fang, *et al*.. Botgraph: large scale spamming botnet detection[C]. Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation, Boston, USA, 2009: 1–14.

[13] Wang Ping, Sparks S, and Zou C. An advanced hybrid Peer-to-Peer botnet[J]. *IEEE Transactions on Dependable and Secure Computing*, 2010, 7(2): 113–127.

[14] 方滨兴, 崔翔, 王威. 僵尸网络综述[J]. 计算机研究与发展, 2011, 48(8): 1315–1331.

[15] Wang Ping, Wu Lei, Aslam B, *et al*.. A systematic study on Peer-to-Peer botnets[C]. Proceedings on Computer Communications and Networks, San Francisco, CA, USA, 2009: 1–8.

[16] Yen Ting-fang. Detecting stealthy malware using behavioral features in network traffic[D]. [Ph.D. dissertation], Carnegie Mellon University(USA), 2011.

[17] DARPA Intrusion Detection Data Sets[OL]. http://www. ll. mit.edu/mission/communications/ist/corpora/ideval/data/ index.html, 2012. 3.

[18] 钱进, 苗夺谦, 张泽华. 云计算环境下知识约简算法[J]. 计算机学报, 2011, 34(12): 2332–2343.

蒋鸿玲： 女，1986 年生，博士生，研究方向为网络安全与云计算.

邵秀丽： 女，1963 年生，教授，博士生导师，研究方向为云计算、网络安全、数据挖掘.

李耀芳： 女，1978 年生，讲师，研究方向为网络安全、数据挖掘.