一种多协议网络日志二次聚类方法研究*

杨 杰
(湖南科技学院 计算机与通信工程系, 湖南 永州 425100)

摘 要: 为了处理网络日志规模过大及其相关问题,并为后期日志分析提供简洁的数据源,提出一种多协议网络日志二次聚类方法。该方法采用划分网格的方式把网络日志进行网格内初次聚类,然后再依据相似度判断对初次聚类簇进行二次聚类,最后输出聚类后的日志记录及一些稀疏数据和孤立点数据。经实验测试证明,在不破坏网络日志的完整性和准确性,且不影响用户正常网络访问的前提下,该方法日志规模压缩效果显著,时间复杂度低以及能够处理实际的动态数据,实现增量式聚类。

关键词: 网络日志分析; 网格聚类; 二次聚类; 增量式聚类
中图分类号: TP309 **文献标志码**: A **文章编号**: 1001-3695(2012)10-3929-03
doi:10.3969/j.issn.1001-3695.2012.10.087

Method of multi-protocol network log two-step clustering

YANG Jie
(*Dept. of Computer & Communication Engineering, Hunan University of Science & Engineering, Yongzhou Hunan* 425100, *China*)

Abstract: To deal with large scale of Web log and related issues, and to provide brief data sources for the later log analysis, this paper proposed a method multi-protocol network log two-step clustering, which ploted every log into data grid and first clustering in the grid. Then according to similarity judgment, made the initial cluster grid secondary clustering. Finally output clustered log, some sparse data and outlier data. Through the test experiment, in the premise of ensuring the completeness and accuracy of log, and without affect the normal user network communication, the method can effectively compress log storage, reduce the time complexity and deal with actual dynamic data and realize incremental clustering.

Key words: Web log analysis; grid-based clustering; two-step clustering; incremental clustering

随着全球信息化的加速,互联网在中国的发展规模迅速扩大。但由于互联网具有开放性、互连性、共享性的特点,网络安全问题也越来越严重,计算机的安全、取证工作也变得越来越重要[1]。网络日志是记录用户对服务器访问情况的日志文件[2],这些文件包含了大量的用户访问信息,如用户的 IP 地址、所访问的 URL、访问日期、事件和访问路径等,通过日志可以监控系统资源[3]、快速定位故障、审计用户行为、告警可疑行为、发现攻击者留下的痕迹、为恢复系统提供帮助、生成调查报告、为打击计算机犯罪提供证据来源[4]等。因此,为了保护系统的安全、方便调查系统故障、监控系统运行状况等,通过对日志记录进行分析,可将整个或局部网络的安全态势进行展示[5]。

在 1980 年,HOSSEIN 等人[6]第一次提出了利用日志进行安全审计的思想。密苏里大学的 Meyer 对日志在当今网络环境中的研究意义作了分析[7],他认为日志是分析网络安全必不可少的数据源。目前国外关于日志的研究主要集中在日志统计分析、实时监控和安全集中管理方面[3],国内主要是运用日志的关联分析来提取日志的规律,然后利用日志进行安全审计。目前国内外都在加紧研究适合当今高速网络环境的网络日志处理技术。归纳起来,主要有两大类:a)在重要的网络节点配置大容量硬盘,存储海量日志;b)配置具有海量存储设备的专用日志服务器[8]。以上两种方式虽然从一定程度上缓解了海量日志的存储问题,但没有从根本上解决网络日志的数据量大、规模巨型化的问题,一方面给系统的性能和存储空间带来较重的负担,同时也使得审计系统分析日志数据时延增加[9],进而严重影响系统安全审计、监控和检测的效率。本文针对网络日志数据量大、数据集动态变化等问题,提出了一种多协议网络日志二次聚类方法。

1 二次聚类方法分析

1.1 聚类方法流程设计

如图 1 所示,对到达和通过网络连通设备(网卡)的多协议报文使用 WinPcap 进行捕获和解析,解析出相关属性值。通过特征选择和特征提取,选取对日志聚类有用的属性字段。对选取的属性值作相似度计算,满足条件的日志记录作聚类处理。在聚类过程中,首先在时间维上划分网格,对网格内的数据作初次聚类,生成初次聚类簇;然后把初次聚类簇进行相似度判断,再次聚类;最后生成日志记录,输出为两次聚类生成的日志记录和一些稀疏数据、孤立点数据。

1.2 各模块具体分析

a)生成网格数据。通过 WinPcap 进行捕包解析,获取属性值,将生成的日志记录以秒为单位存储到数据库中,作为原始的数据源。日志文中字符表示如下: msStart 为起始微妙值、msEnd 为结束微妙值、dtStart 为起始时间、dtEnd 为结束时间、cnt 为聚类时聚合的报文段个数、lens 为初次聚类簇传输的数

收稿日期: 2012-01-16; **修回日期**: 2012-03-18 **基金项目**: 湖南省自然科学基金资助项目(11JJ6056)
作者简介:杨杰(1976-),男(瑶族),湖南永州人,副教授,硕士,主要研究方向为计算机网络安全、智能计算(41206991@qq.com).

据量、lens-sd 为从源到目的传输的数据量、lens-ds 为从目的到原传输的数据量、synNum 为连接开始标志、finNum 为连接结束标志。

b) 初次聚类。只选择 sIP、dIP、sPort、dPort 相同的日志记录作为聚类项,并转换生成 msStart、msEnd、lens、cnt(聚合的报文段数)。设计了相应的 SQL 语句:

```
SELECT MAX(datetime) AS dt,
MIN(micsec) AS msEnd;
sIP、dIP、sPort、dPort,
SUM(len) AS lens,COUNT(*) AS cnt,
SUM(syn) AS synNum,SUM(fin) AS finNum,
FROM [以时间值命名的网格数据表]
WHERE GROUP BY sIP、dIP、sPort、dPort
```

经过数据库处理可直接生成聚类簇,且以日志记录的形式存储于数据库的中间临时表(udpMid, tcpMid, icmpMid, eMid, arpMid),同时删除原有的网格数据,以减小数据集的规模,便于进行动态增量式聚类。

c) 二次聚类。每隔一段时间对初次聚类簇网格作二次聚类处理。设有初次聚类簇 $A,X,Y,\cdots$,最终聚类簇表示为 C。

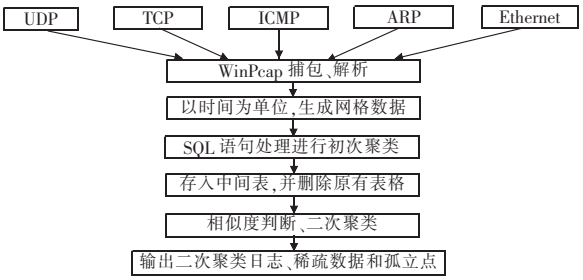

图 1 聚类方法流程设计

1.2.1 UDP 协议二次聚类方法

a) A 作为 C 第一条源数据,直接给 C 各属性赋值:

$C_{dPort}=A_{dPort};C_{SizeSum}=A_{lens}$

b) C 与下一条数据 X 作匹配对比

①if $C_{dtStart}>X_{dt}$

$C_{MicStart}=X_{msStart}$

②if($C_{dtEnd}<X_{dt}$)

$C_{MicEnd}=X_{msEnd}$

③flag = true

if(flag = fase)

$C_{sPort}=X_{sPort};C_{dPort}=X_{dPort};C_{SizeSum}=X_{lens}$

1.2.2 TCP 协议二次聚类方法

a) A 作为 C 第一条源数据,直接给 C 各属性赋值:

$C_{synNum}=A_{synNum}$

b) C 与下一条数据 X 作匹配比对:

if($C_{dtEnd}-X_{dt}\leqslant 3(min)\parallel X_{dt}-C_{dtStart}\leqslant 3(min)$)

if(flag = false)

$C_{dip}=X_{dip};C_{sPort}=X_{sPort};C_{dPort}=X_{dPort}$

c) C 作为最终结果存入数据库 Clus。

1.2.3 ICMP 协议二次聚类方法

a) A 作为 C 第一条源数据,直接给 C 各属性赋值:

$C_{dip}=X_{dip};C_{typ}=A_{dyp};C_{code}=A_{code}$

①if($C_{dtStart}>X_{dt}$)

$C_{MicStart}=X_{msStart}$

②if($C_{dtEnd}<X_{dt}$)

$C_{MicEnd}=X_{msEnd}$

③flag = true

if(flag = false)

$C_{sPort}=X_{sPort};C_{dPort}=X_{dPort};C_{SizeSum}=X_{lens}$

b) C 作为最终结果存入数据库 Clus 中。

1.2.4 ARP 协议二次聚类方法

a) A 作为 C 第一条源数据,直接给 C 各属性赋值:

$C_{dMac}=A_{dMac}$

b) C 与下一条数据 X 作匹配比对:

if($C_{dtStart}>X_{dt}$)

$C_{MicStart}=X_{msStart}$

if($C_{dtEnd}<X_{dt}$)

$C_{MicEnd}=X_{msEnd}$

③flag = true

if(flag = flase)

$C_{dMac}=X_{sMac};C_{dMac}=X_{dMac}$

c) C 作为最终结果存入数据库 Clus。

1.2.5 Ethernet 协议二次聚类方法

a) A 作为 C 第一条源数据,直接给 C 各属性赋值:

$C_{dip}=A_{dip};C_{sMac}=A_{sMac};C_{dMac}=A_{dMac};C_{proto}=A_{proto}$

b) C 与下一条数据 X 作匹配比对:

if($X_{dt}-C_{dtEnd}<3(min)$&&($C_{sMac}=X_{sMac}$&&$C_{dMac}=X_{dMac}$))

①if($C_{dtStart}>X_{dt}$)

$C_{MicStart}=X_{msStart}$

②if($C_{dtEnd}<X_{dt}$)

$C_{MicEnd}=X_{msEnd}$

③flag = true

if(flag = false)

$C_{dip}=X_{dip};C_{sMac}=X_{sMac};C_{dMac}=X_{dMac};C_{proto}=X_{proto}$

2 实验与结果分析

2.1 实验定性分析

文中设计的实验程序是在 Visual Studio 2005 平台上实现的一个简单的控制台程序。在整个程序实现过程中,为降低对内存和 CPU 的占用率,虽然界面没有显示数据处理过程,但由于多次用到了数据库相关技术,从整体来说加大了对内存和 CPU 的耗费。然而从效果上来说减轻了程序本身的负担,降低了时间和空间复杂度,同时满足网络日志准确性和完整性的基本要求。从操作上来说,捕包解析、存储生成网格、初次聚类和二次聚类间相互独立,同时运用多线程技术,提高了程序的运行效率。

2.2 实验定量分析

实验从 2011-3-15 18:20:11 到 2011-3-15 21:30:45,持续 3:10:34,其中有数据流的高峰期,也有稀疏数据的时间段。因为数据量较多,表 1 只展示了实验的部分数据。

从表 1 中可以清楚地看到日志压缩效果显著。以 TCP 为例,2011-3-15 18:20:11 至 2011-3-15 18:20:20 期间,源 IP(sIP)为 123.125.115.59,源端口(sPort)为 80 与目的 IP(dIP)为 192.168.0.97,目的端口(dPort)为 4 668 之间的通信量为从源端到目的端(size_sd)为 25 472 条,从目的端到源端(size_ds)为 1 301 条,但经过二次聚类之后日志数(cnt)约减为 34 条。在整个实验中,流经网络互连设备的报文数共计 3 582 050,会占用 1 857 MB 的内存,经聚类后只产生 224 546 条日志,占用 239 MB 的内存。

表 1　二次聚类部分结果

dtStart	msStart	dtEnd	msEnd	Proto	sIP	dIP	sPort	dPort	cnt	size_sd	size_ds	rst	syNum	finNum
2011-3-15 18:20:11	135290	2011-3-15 18:20:20	705196	TCP	123.125.115.59	192.168.0.97	80	4668	34	25472	1301	1	2	1
2011-3-15 18:20:11	338327	2011-3-15 18:20:11	955543	TCP	192.168.0.97	222.68.192.43	4671	80	16	692	12116	0	2	1
2011-3-15 18:20:11	114917	2011-3-15 18:20:45	905695	TCP	192.168.0.97	60.210.18.164	4667	80	11	1037	745	1	2	1
2011-3-15 18:20:15	845545	2011-3-15 18:19:16	51154	TCP	192.168.0.97	202.97.246.219	4595	80	26	2971	12513	2	2	0
2011-3-15 18:20:19	590232	2011-3-15 18:20:20	724499	TCP	192.168.0.97	218.78.215.78	4674	80	23	2786	5628	2	2	0
2011-3-15 18:20:27	669012	2011-3-15 18:19:57	681088	TCP	123.129.214.147	192.168.0.97	80	4630	14	5174	738	1	2	1
2011-3-15 18:20:29	739086	2011-3-15 18:20:45	905697	TCP	123.125.115.75	192.168.0.97	80	4687	35	14273	7298	1	2	1
2011-3-15 18:20:31	974415	2011-3-15 18:20:50	951917	TCP	119.164.219.6	192.168.0.97	80	4698	69	58081	3043	1	2	1
2011-3-15 18:20:30	974416	2011-3-15 18:20:50	951912	TCP	119.164.219.6	192.168.0.97	80	4700	171	173698	4677	1	2	1
2011-3-15 18:34:00	221051	2011-3-15 18:37:12	732459	UDP	192.168.0.97	119.162.80.43	57624	2627	149	93036	NULL	NULL	NULL	NULL
2011-3-15 18:34:06	211729	2011-3-15 18:34:06	921161	UDP	192.168.0.97	119.177.203.250	57624	1882	8	5328	NULL	NULL	NULL	NULL
2011-3-15 18:34:06	188049	2011-3-15 18:36:16	324865	UDP	192.168.0.97	123.232.154.57	57624	2160	24	11248	NULL	NULL	NULL	NULL
2011-3-15 18:33:14	998122	2011-3-15 18:36:17	235535	UDP	192.168.0.97	182.33.109.182	57624	4033	60	40944	NULL	NULL	NULL	NULL
2011-3-15 18:33:15	7055	2011-3-15 18:36:17	305829	UDP	192.168.0.97	221.2.76.34	57624	12701	91	47828	NULL	NULL	NULL	NULL
2011-3-15 18:34:02	218859	2011-3-15 18:34:06	845051	UDP	192.168.0.97	60.208.151.54	57624	2516	14	9016	NULL	NULL	NULL	NULL
2011-3-15 18:33:15	25561	2011-3-15 18:36:17	288981	UDP	192.168.0.97	60.215.102.114	57624	1575	55	32772	NULL	NULL	NULL	NULL
2011-3-15 18:33:15	631029	2011-3-15 18:33:43	125397	ARP	192.168.0.97	192.168.0.11	f0:de:f1:0	00:00:00:0	1	3	NULL	NULL	NULL	NULL
2011-3-15 18:33:50	333443	2011-3-15 18:37:14	444407	ARP	192.168.0.1	192.168.0.102	00:e0:4c:3	00:00:00:0	1	17	NULL	NULL	NULL	NULL
2011-3-15 18:36:30	594659	2011-3-15 18:37:14	585859	ARP	192.168.0.97	192.168.0.101	f0:de:f1:0	00:00:00:0	1	5	NULL	NULL	NULL	NULL
2011-3-15 19:05:30	432758	2011-3-15 19:09:27	582332	ARP	192.168.0.1	192.168.0.102	00:e0:4c:3	00:00:00:0	1	43	NULL	NULL	NULL	NULL
2011-3-15 19:05:33	983961	2011-3-15 19:09:25	985340	ARP	192.168.0.1	192.168.0.8	00:e0:4c:3	20:6a:8a:0	2	7	NULL	NULL	NULL	NULL
2011-3-15 19:05:30	266893	2011-3-15 19:08:30	774323	ARP	192.168.0.1	192.168.0.167	00:e0:4c:3	00:26:2d:3	2	5	NULL	NULL	NULL	NULL
2011-3-15 19:20:33	168618	2011-3-15 19:25:31	661095	ICMP	192.168.0.138	210.44.3.254	8	0	172	NULL	NULL	NULL	NULL	NULL
2011-3-15 19:23:00	385922	2011-3-15 19:23:00	385922	ICMP	58.17.163.48	192.168.0.169	3	10	1	NULL	NULL	NULL	NULL	NULL
2011-3-15 19:22:36	317751	2011-3-15 19:22:45	314824	ICMP	202.205.13.238	192.168.0.169	11	0	6	NULL	NULL	NULL	NULL	NULL
2011-3-15 19:22:18	920102	2011-3-15 19:22:18	920102	ICMP	192.168.101.254	192.168.0.138	3	1	1	NULL	NULL	NULL	NULL	NULL
2011-3-15 19:39:09	322689	2011-3-15 19:39:12	327860	ICMP	60.210.0.128	192.168.0.138	3	13	2	NULL	NULL	NULL	NULL	NULL
2011-3-15 20:17:35	3179	2011-3-15 20:21:34	997088	Ethernet	00:1b:fc:0a:1a:85	00:e0:4c:3a:1b:87	2048	NULL	22928	NULL	NULL	NULL	NULL	NULL
2011-3-15 20:17:36	342238	2011-3-15 20:21:34	688160	Ethernet	00:26:2d:36:cb:fc	00:e0:4c:3a:1b:87	2048	NULL	2142	NULL	NULL	NULL	NULL	NULL
2011-3-15 20:17:37	83446	2011-3-15 20:21:31	808263	Ethernet	00:26:2d:36:cc:46	00:e0:4c:3a:1b:87	2048	NULL	157	NULL	NULL	NULL	NULL	NULL
2011-3-15 20:17:38	785171	2011-3-15 20:21:33	564385	Ethernet	00:26:2d:37:87:33	00:e0:4c:3a:1b:87	2048	NULL	456	NULL	NULL	NULL	NULL	NULL
2011-3-15 20:18:21	801163	2011-3-15 20:21:21	800067	Ethernet	00:26:2d:36:cc:46	01:00:5e:7f:ff:fa	2048	NULL	20	NULL	NULL	NULL	NULL	NULL

该聚类方法时间复杂度低。因为采用基于网格的二次聚类,其时间复杂度主要取决于以下两个方面:a)初次聚类取决于数据规模,时间复杂度为 $O(n)$(n 为网格内数据量);b)二次聚类取决于中间临时表数据库(如 udpMid、tcpMid、icmpMid、eMid、arpMid)的规模,因为聚类过程中有些数据已经被聚合,不必都参与遍历,其时间复杂度低于 $O(n^2)$(n 为初次聚类簇个数)。

由于两次聚类操作都以多线程的方式并行运作,不影响日志的捕获、解析,使得程序可以不断地向数据集中添加数据,实现增量式聚类;同时两次聚类过程中都设置了数据的删除环节,不会使整个数据集规模过大。因此,该聚类模型适合于动态的实时数据,完全不同于以往传统的面对静态数据集的聚类算法。

3　结束语

针对网络日志数量大、数据集动态变化等问题,提出了一种基于网格的多协议网络日志二次聚类方法。该方法采用划分网格的方式把网络日志进行网格内初次聚类,然后再依据相似度判断对初次聚类簇进行二次聚类,实现了对海量日志的有效压缩,程序简单,易于实现,时间复杂度低。同时结合多线程技术和数据库技术,提高了程序的运行效率,为快速定位故障、监控用户的行为、发现异常情况以及发现攻击者留下的痕迹等问题提供了简洁的数据源。另外,因为多次用到数据库相关技术,从总体上加大了对内存和 CPU 的耗费,所以选择适当方法来减少对数据库的操作是下一步要进行的工作。

参考文献:

[1] CAO Sha-sha, KE Zong-wu, CHEN Nian-sheng. Research of QoS routing technology for wireless multimedia sensor network[C]//Proc of ISECS International Colloquium on Computing, Communication, Control, and Management. 2009:334-337.

[2] ZHANG Xia, YU Hong-yi, ZHOU Gang. Bandwidth efficient collaborative quality of service routing for real-time flow in wireless multimedia sensor networks[C]//Proc of Services Computing Conference. 2010: 509-515.

[3] SUN Wen-zhong, SONG Ying-xiong, CHEN Min. A load-balanced and energy-aware routing metric for wireless multimedia sensor networks [C]//Proc of the 3rd International Conference on Wireless, Mobile and Multimedia Networks. 2010:21-24.

[4] MEDJIAH S, AHMED T, KRIEF F. AGEM: adaptive greedy-compass energy-aware multipath routing protocol for WMSNs[C]//Proc of the 7th IEEE Consumer Communications and Networking Conference. 2010: 1-6.

[5] MOHAJERZADEH A H, YAGHMAEE M H, MONSEFI R. A QoS based data dissemination protocol for wireless multimedia sensor networks[C]//Proc of the 3rd International Workshop on Advanced Computational Intelligence. 2010:670-675.

[6] HOSSEIN M A, HOSSEIN Y M, NAJMI T N, *et al*. MREEP: a QoS based routing protocol for wireless multimedia sensor networks[C]// Proc of the 19th Iranian Conference on Electrical Engineering (ICEE). 2011:1-6.

[7] VITHYA G, VINAYAGASUNDARAM B. Actuation sensor with adaptive routing and QOS aware checkpoint arrangement on Wireless Multimedia Sensor Network[C]//Proc of International Conference on Recent Trends in Information Technology. 2011:444-449.

[8] 赵小敏. 基于日志的计算机取证技术的研究及系统设计与实现[D]. 浙江:浙江工业大学信息工程系,2008.

[9] PETER K K L, HSU W J, YI P. Performance evaluation of efficient and reliable routing protocols for fixed-power sensor networks [J]. IEEE Trans on Wireless Communications, 2009, 8(5): 1102-1109.

[10] 刘必雄,杨泽明,吴焕,等. 基于集群的多源日志综合审计系统[J]. 计算机应用,2008,28(2):541-544.

[11] 卿斯汉,温红子,雷浩,等. 基于 Clark-Wilson 完整性策略的安全监视模型[J]. 软件学报,2004,15(8):1124-1132.