高技术通讯 2010 年 第 20 卷 第 10 期:1033 ~ 1039

doi:10.3772/j.issn.1002-0470.2010.10.008

基于二值异或群的多轨迹识别算法①

朱洪亮②*** 李 锐*** 辛 阳***** 杨义先*** 徐国爱***

(*北京邮电大学网络与信息攻防技术教育部重点实验室 北京 100876)
(**北京邮电大学网络与交换技术国家重点实验室信息安全中心 北京 100876)
(***北京安码科技有限公司 北京 100876)

摘 要 针对二值异或群循环轨迹集没有有效的识别算法的问题,进行了高效快速的多轨迹识别算法方面的研究。首先建立了多轨迹识别问题的数学模型,定义了一类新的轨迹——二值异或群循环轨迹,然后针对这类轨迹提出了一种新的识别算法,将求解与基轨迹相异或的元素值的问题转化为求解轨迹特征位模式的问题,并在理论上证明了其正确性,通过这样的处理极大地提高了算法可操作性,最后将此算法应用于共享接入检测场景中并进行了验证。实验证明,此算法具有较高的准确度和误报容忍度,能较好地应用于网络监控和信息安全领域。

关键词 多轨迹识别,二值异或群,特征位,共享接入,网络监控

0 引 言

随着下一代网络(next generation network, NGN)、第三代移动通信(3rd generation, 3G)等电信技术的发展,运营商的建网成本逐步降低,电信业务部署越来越灵活,新兴运营商可以很容易加入市场中。同时,运营商对网络应用逐渐失去掌控能力,监管不到位,致使电信业务在运营过程中出现许多灰色地带。当前无监管的网络电话(voice over IP, VoIP)业务、对等网络(peer to peer, P2P)业务[1,2]以及不受控的宽带私接冲击着各大运营商的运营管理模式。针对这种现状,基于业务的精细化运营是大势所趋。从业务监控、初级运营到高级阶段,运营商需要详细的流量分析和用户行为数据,为流量管理和业务运营提供管理手段,并提供定制化服务,改进运营手段,从而提高业务收入。

宽带数据业务是拉动运营商业务增长的重要组成部分,无序共享上网给运营商产生了诸多不利,如运营成本增加、用户流失、合法用户受到冲击、账号盗用等,因此急需能够对共享上网主机进行精确检测的算法并继而实现控制来降低这些风险。本文研究了域名系统(domain name system, DNS)的主机变化特征,建立了二值异或群(binary XOR group)循环轨迹识别的数学模型,通过理论证明将轨迹异或问题转化为确定轨迹模式的问题,从而简化了检测处理流程,提供了一种高效易用的检测算法,适用于二值异或群循环轨迹集的检测,自然也可应用于共享接入主机数目的检测。最后对算法进行了特性分析并给出实验结果。

1 相关工作

目前多轨迹识别已在很多领域中涉及到,尤其在网络监控和网络安全领域,主要指如何将混合在一起的具有类似特性的多条轨迹链数据区分出来。粗放型要求是识别轨迹链数量,精细化要求是定位出每条轨迹链的数据满足细粒度需要。最典型的应用就是共享接入主机数目的检测以及特征指纹识别。当前的主要算法有 Bellovin 提出的网络地址翻译(network address translation, NAT)主机数检测算法[3],该算法提出利用 IP 协议标识(IPID)[4]的递增规律进行检测,并采用时间切割方法获取主机数,但目前很多 NAT 设备都能修改 IPID 值,从而规避该算法。还有 Kohno 提出的基于传输控制协议(transmission control protocol, TCP)时间戳[5]的主机数目检测

① 863 计划(2009AA01Z439)和国家自然科学基金(60821001,U0835001)资助项目。
② 男,1982 年生,博士;研究方向:信息安全,网络监控等;联系人,E-mail: zhuhongliang_82@yahoo.com.cn
(收稿日期:2009-07-14)

算法[6],该算法利用TCP时间戳选项进行检测,准确性较高,但TCP时间戳需要进行主动触发,会影响正常用户行为,同时也易于被网络地址翻译(NAT)设备修改。其他一些算法[7,8]利用主机特性或行为特征进行检测,区分不同轨迹,这类检测算法准确性不高,并且需要大量样本,可靠性不强。此外,目前这些算法并没有理论基础,都偏向于工程实现,不利于一般问题的推广。

2 基本概念

设$(Z_n,+)$为模n剩余类群[9],$(Z_2^n,\oplus)$为二值n元异或群[10],$g=(c_1,c_2,\cdots,c_n)(c_i=0$或$1)\in Z_2^n$。$Z_n$和$Z_2^n$可看作同一集合的不同表示方式,如$Z_{16}$和$Z_2^4$,$Z_{16}$可看作十进制表示$\{0,1,2,\cdots,15\}$,其对应的二进制表示$\{0000,0001,0010,\cdots,1111\}$即为$Z_2^4$。

定义1:称n元有序组$T=(t_1,t_2,\cdots,t_n)$为n元轨迹,满足:$\forall i,j\in[1,n]\cap N$,若$i\neq j$且$t_i\neq t_j$,则$(t_1,\cdots,t_i,\cdots t_j,\cdots,t_n)\neq(t_1,\cdots,t_j,\cdots t_i,\cdots,t_n)$。其中$t_i\in C$($C$为集合),称$t_1$为第一分量,$t_2$为第二分量,$\cdots,t_n$为第$n$分量。若$n$元轨迹$T_1=T_2$,当且仅当$\forall i\in[1,n]\cap N$都有第$i$分量相等。

定义2:设$T_1=(t_1,t_2,\cdots,t_n)$为n元轨迹,则称轨迹$T_2=(t_1,t_2,\cdots,t_n,t_{n+1},\cdots,t_{2n},\cdots)$为$n$元循环轨迹,满足:$\forall i,j\in N$,若$i\equiv j(\mathrm{mod}n)$,则$t_i=t_j$,同时不存在$m<n$,使得$\forall i,j\in N$,若$i\equiv j(\mathrm{mod}m)$,则$t_i=t_j$恒成立。如果集合$G=\{t_1,t_2,\cdots,t_n\}$对于某代数运算。构成$n$元群,则称$T_2$为$n$元群循环轨迹。典型$n$元群循环轨迹如模$n$剩余类群循环轨迹$T(Z_n)=([0],[1],\cdots,[n-1],[0],[1],\cdots,[n-1],\cdots)$(称为标准模$n$剩余类群循环轨迹)。类似地,标准二值$n$元异或群循环轨迹

$T(Z_2^p)=(\underbrace{00\cdots00}_{p个},\underbrace{00\cdots01}_{p个},\underbrace{0\cdots010}_{p个},\cdots,\underbrace{11\cdots1}_{p个},\cdots)$(其中$n=2^p$)。

定义3:设$T=(t_1,t_2,\cdots,t_n,\cdots)$为$n$元群循环轨迹,若$i\in[1,n]\cap N$,则$T\circ t_i=(t_1\circ t_i,t_2\circ t_i,\cdots,t_n\circ t_i,\cdots)$。

由定义知,轨迹具有序列性,现实中比较显著的一类轨迹为时间序列循环轨迹,由时间先后性决定了分量的位置差异性,记作$S(t)=(s(t_1),s(t_2),\cdots,s(t_n),\cdots)$,其中$t_1<t_2<\cdots<t_n<\cdots,t_1$为起始时间点。

定义4:循环轨迹集$S=\{S_i(t)\mid S_i(t)=T(Z_2^p)\oplus k_i$,其中$T(Z_2^p)$为一固定二值异或群循环轨迹,$n=2^p,k_i\in Z_2^p,i=1,2,\cdots m\}$称为时间序列二值异或群循环轨迹集,称固定的二值异或群循环轨迹$T(Z_2^p)$为基轨迹。若基轨迹$T(Z_2^p)$为标准二值n元异或群循环轨迹,则称S为标准时间序列二值异或群循环轨迹集。S集合中各轨迹共用相同时间轴,因此,相邻时间的分量可能属于不同的轨迹$S_i(t)$和$S_j(t),i,j\in[1,m]\cap N$。S集合中各元素轨迹为同一二值异或群轨迹与不同群元素相异或的结果。由定义知,S中元素个数$\#S=m$。

在严格意义下,对时间序列二值n元异或群循环轨迹$S(t)$,当前时刻t_{cur}对应取值,则下一时刻t_{next}取值$S(t_{\mathrm{next}})=S(t_{\mathrm{cur}+1})$。现实环境中总存在一些异常情况,下面定义一种异常情况。

定义5:对于时间序列二值n元循环轨迹$S(t)$,在同一周期$T(T\in[kn+1,kn+n],k\in N)$内,当前时刻$t_{\mathrm{cur}}$对应取值,若下一时刻$t_{\mathrm{next}}$对应元素$S=S(t_{\mathrm{rear}})$,其中$t_{\mathrm{rear}}>t_{\mathrm{cur}+1}$,则称为跳变。

本文重点研究标准时间序列二值异或群循环轨迹集S的多轨迹识别问题。

3 多轨迹识别算法

3.1 问题提出

为解决网间协议第四版(Internet protocol version 4, IPv4)地址资源匮乏的问题,Internet工程任务组(Internet Engineering Task Force, IETF)组织提出了网络地址翻译[11](NAT)技术。NAT设备主要完成地址转换的功能,除了解决IPv4地址不足的问题外还能起到隐藏并保护内部网络的作用。当前最常用的一种实现方式采用端口多路复用,位于NAT后的主机拥有自己的私网IP地址段如C类地址192.168.10.1,这些地址不能直接访问互联网;而NAT设备则拥有一个合法的公网IP地址,通过私网地址和该IP不同端口的映射实现内部网络所有主机对Internet的访问。由此可见,不同的NAT后主机在公网设备看来具有相同的公网IP地址,因此相对公网设备并不透明,造成共享上网无序性和潜在运营风险,为达到一定监管目的,很多情况下需要利用主机特有属性对NAT后的主机数量进行检测,DNS事务号的变化规律

就属于一类特有主机属性。

DNS是域名系统的缩写,相应规范和标准由IETF制定[12,13]。在Internet上域名与IP地址之间是一对一(或者一对多)的,域名虽然便于人们记忆,但机器之间只能互相认识IP地址,它们之间的转换工作称为域名解析,域名解析需要由专门的域名解析服务器来完成,DNS就是进行域名解析的服务器。DNS协议中定义了一个既可以查询也可以响应的报文格式。协议规范中前两个字节为Transaction ID字段(记为TransID),表示事务号,该字段唯一标识同一事务,即针对同一查询的请求和响应拥有相同的TransID值,而不同的查询和响应则对应不同的TransID值。

在Windows系统中[14],"DNS客户(Client)"服务用于缓存DNS响应,当其运行时用户数据报协议(user datagram protocol, UDP)源端口是静态的,服务启动时获得,服务终止时释放。但源端口也不是一直固定,会有一定的变动。如果在操作系统生命周期内重启该服务,就会分配一个新的UDP源端口。当DNS Client服务未开启时,UDP端口一般是递增的。TransID的生成主要由DNS Client服务决定,通过公式

$$S_n = (S_{n-1} + ((n + \tau)\bmod 487) + 1)\bmod 2^{16} \quad (1)$$

$$TransID = S_n \oplus K \quad (2)$$

产生。其中:n是一个全局计数器,每发出一个针对新域名的DNS查询该值增1;τ是客户端时间,以时钟周期为单位,对单核CPU来说,时钟周期大概持续10.0144ms;S_n为16位的中间变量;K由"DNS Client"服务在启动时确定,如果重启,则分配新的K值;$\oplus$表示异或运算。DNS数据包中的TransID为(2)式TransID值的低字节序(低位在前,高位在后)排列。由(1)、(2)式可知:

$$S_n - S_{n-1} = TransID_n \oplus K - TransID_{n-1} \oplus K = ((n + \tau)\bmod 487 + 1)\bmod 2^{16} \quad (3)$$

由(3)式知,因为$((n + \tau)\bmod 487 + 1)\bmod 2^{16} < 488$,因此$S_n$高八位增幅很小(小于2),基本固定,即TransID的高位字节与某固定值K异或之后基本递增(当增幅取2时就出现跳变情况),该递增序列实际上就是标准二值n元异或群循环轨迹($n = 2^8$),因为K对同一主机是固定的,所以TransID的高位字节的轨迹就是标准二值异或群循环轨迹作为基轨迹与K异或的结果,为二值异或群循环轨迹。在数据报格式中就是两个字节的后一字节(称为特征位)与某固定值异或之后递增。在实验数据中观察到的变化规律确实如此,尤其特征位是4位时,变化周期短,规律明显。该规律适用于Windows XP SP2系统,更新MS08-020公告补丁之前[15]。同时,也适用于Windows Vista系统和Windows 2003系统,区别主要体现在n值的不同上,而特征位的变化规律没有区别。

因为不同主机分配的K值不同,因此其DNS TransID具有不同的变化规律,通过统计NAT后各主机发出的DNS查询数据包的TransID的变化情况来达到检测NAT后主机的目的。而对不同变化规律的识别重点需要分辨出不同的异或值,使用传统的遍历方式效率较低,并且比较复杂,检测周期长,不利于工程实现,需要研究一种简单快捷高效的检测算法。不难发现,每台主机的DNS TransID就构成二值异或群循环轨迹,而NAT后主机数目的检测就属于时间序列二值异或群循环轨迹集的多轨迹识别问题。

3.2 理论基础

定理1:任何二值循环轨迹$T(G) = T(Z_2^p) \oplus k$,$k \in Z_2^p$,$n = 2^p$为异或群循环轨迹。

证明:设$T(Z_2^p) = (t_1, t_2, \cdots, t_n, \cdots)$,则$T(G) = (t_1 \oplus k, t_2 \oplus k, \cdots, t_n \oplus k, \cdots)$。

因为$Z_2^p = \{t_1, t_2, \cdots, t_n\}$,$(Z_2^p, \oplus)$为$n$阶群。令$G = \{t_1 \oplus k, t_2 \oplus k, \cdots, t_n \oplus k\}$,若证$T(G)$为异或群循环轨迹,只需证明:$Z_2^p = G$。

(1) $\forall t_i \in Z_2^p$, $i \in [1, n] \cap N$,因为$k \in Z_2^p$,所以$t_i \oplus k \in Z_2^p$,因为$\exists j \in [1, n] \cap N$,使得$t_j = t_i \oplus k$;所以$t_j \oplus k \in G$,即$t_j \oplus k = t_i \oplus k \oplus k = t_i \in G$。

(2) $\forall i \in [1, n] \cap N$,$t_i \oplus k \in G$,因为$t_i \in Z_2^p$,$k \in Z_2^p$,所以$t_i \oplus k \in Z_2^p$。

所以$Z_2^p = G$,$T(G)$为二值异或群循环轨迹,得证。

对于二值n元异或群循环轨迹$T(G) = T(Z_2^p) \oplus k = (t_1, t_2, \cdots, t_n, \cdots)$,$k \in Z_2^p$,$n = 2^p$,如果对于任意连续元素$t_i, t_{i+1}, i \in [1, +\infty] \cap N$,$t_i = (c_1, c_2, \cdots c_p)$,$t_{i+1} = (c_1', c_2', \cdots, c_p')$,$(c_i, c_i' = 0 \text{或} 1)$,前$j(1 < j < p)$位均相等,即$c_1 = c_1'$, $c_2 = c_2'$, $\cdots$, $c_j = c_j'$,第$j+1$位不等,都有$c_{j+1} = 0, c_{j+1}' = 1$,则称$T(G)$第$j+1$位模式为正序;反之,若都有$c_{j+1} = 1$,$c_{j+1}' = 0$,则称为反序。若$T(G)$除第1位之外(第1位前面没有位可以参照)的所有位均有序(正序或反序),则称$T(G)$是有序的,可记作$[\lambda_2, \lambda_3, \cdots, \lambda_p]$($\lambda_i = +$或$-$),$+$表示正序,$-$表示反序。标准二值$n$元异或群循环轨迹$T(Z_2^p) = (\underbrace{00\cdots00}_{p\text{个}}, \underbrace{00\cdots01}_{p\text{个}}, \underbrace{0\cdots010}_{p\text{个}},$

…,$\underbrace{11\cdots1}_{p个}$,…) 即为有序的,序列模式为[+,+,…,+](各位均为正序)。如 $T(Z_2^4)$ ($n=16$)序列模式为[+,+,+],表1中竖线显示每位序列变化模式。

表1 $T(Z_2^4)$ 元素序列模式

位置	元素组	位置	元素组
1	0000	9	1000|
2	0001	10	100|1|
3	0010	11	101|0
4	0011	12	10|11
5	0100	13	11|00
6	0101	14	1101
7	0110	15	1110
8	0111	16	1111

定理2:如果 $T(Z_2^p)$ 是有序的,那么任何二值异或群循环轨迹 $T(G)=T(Z_2^p)\oplus k$, $k\in Z_2^p$, $n=2^p$ 均是有序的。

证明:由上知 $Z_2^p=\{t_1,t_2,\cdots,t_n\}$ 和 $G=\{t_1\oplus k,\ t_2\oplus k,\cdots,t_n\oplus k\}$ 是相等的,同时 $T(Z_2^p)$ 是有序的。

令 $k=(c_1,c_2,\cdots,c_p)$, 其中 $c_i=0$ 或1。若证 $T(G)$ 是有序的,只需要证 $T(G)$ 除第一位之外的所有位均有序。

$\forall i\in[2,p]\cap N$, 对于 $T(G)$ 的第 i 位,因为对于任意连续元素 $t_j\oplus k$, $t_{j+1}\oplus k$, $j\in[1,+\infty]\cap N$, $t_j\oplus k=(\alpha_1,\alpha_2,\cdots,\alpha_p)$, $t_{j+1}\oplus k=(\beta_1,\beta_2,\cdots,\beta_p)$, $(\alpha_i,\beta_i=0$ 或1),若前 $i-1(1<i<p+1)$ 位均相等,第 i 位不等,即 $\alpha_1=\beta_1,\alpha_2=\beta_2,\cdots,\alpha_{i-1}=\beta_{i-1},\alpha_i\neq\beta_i$, 则同样有 $\alpha_1\oplus c_1=\beta_1\oplus c_1,\alpha_2\oplus c_2=\beta_2\oplus c_2,\cdots,\alpha_{i-1}\oplus c_{i-1}=\beta_{i-1}\oplus c_{i-1}$, $\alpha_i\oplus c_i\neq\beta_i\oplus c_i$, 因为 $\alpha_1\oplus c_1$, $\alpha_2\oplus c_2,\cdots,\alpha_i\oplus c_i\in t_j;\beta_1\oplus c_1$, $\beta_2\oplus c_2,\cdots,\beta_i\oplus c_i\in t_{j+1}$, 由 $T(Z_2^p)$ 是有序的定义,都有: $\alpha_i\oplus c_i=0;\beta_i\oplus c_i=1$。

(1) 若 $c_i=0$, 则 $\alpha_i=\alpha_i\oplus c_i\oplus c_i=0\oplus0=0,\beta_i=\beta_i\oplus c_i\oplus c_i=1\oplus0=1$, 所以第 i 位是正序。

(2) 若 $c_i=1$, 则 $\alpha_i=\alpha_i\oplus c_i\oplus c_i=0\oplus1=1,\beta_i=\beta_i\oplus c_i\oplus c_i=1\oplus1=0$, 所以第 i 位是反序。

所以第 i 位是有序的。

由 i 的任意性知, $T(G)$ 是有序的,得证。

因标准二值异或群循环轨迹所有位均为正序,由定理2知,标准二值异或群循环轨迹同某元素异或的结果可以归结为求该二值异或群循环轨迹的序列模式。DNS TransID 就是标准二值异或群循环轨迹作为基轨迹与 K 异或的结果,所以同样适用。对于不同的 $k\in Z_2^p$, 则不同的 $T(G)=T(Z_2^p)\oplus k$ 有不同的模式,除去第一位的模式无法确定,则对于不同的二值异或群循环轨迹共有 2^{p-1} 种模式。

通过确定每一位的序列模式可最终确定二值异或群循环轨迹的模式,如果事先知道两个二值异或群循环轨迹的元素值,判断他们分别是由标准二值异或群循环轨迹与不同元素值异或的过程等同于判断两个循环轨迹的模式,并且如果判断两个循环轨迹的不同,不需要求解出与标准二值异或群循环轨迹异或的元素值的不同,只需要得出两个二值异或群循环轨迹的模式不同,更进一步,只需求出两个二值异或群循环轨迹的某一位的模式不同即可。这样极大地提升了检测两轨迹不同的效率,同时降低了检测算法的复杂度。

3.3 算法流程

算法通过 DNS TransID 的二值异或群循环轨迹规律检测共享接入主机数来显示二值异或群循环轨迹的识别方法。检测系统获取经过 NAT 的 DNS 查询数据包,并提取其中的 TransID 字段。因为同一源 IP 同一源端口的数据必然属于同一台主机的数据,检测系统按照相同源 IP 相同源端口将 TransID 进行归类,对比同一类相邻 TransID 来判断每一位的模式,直至所有位模式判断完,合并具有相同模式的不同端口的 TransID 记录,根据最终得出的模式数判断主机数目。图1为利用 DNS TransID 轨迹检测共享接入主机数的流程。

算法检测流程的详细描述如下:

步骤1:数据采集流程。检测系统位于 NAT 设备出口处,能捕获所有流经 NAT 设备的数据包,过滤出其中的 DNS 查询数据包,从负载开始提取两字节即 TransID 字段,从而得到特征字节,同时获取该 DNS 数据包的源 IP 和源端口信息。

步骤2:轨迹链更新流程。定位出具有相同源 IP 和源端口的轨迹链,如果该轨迹链还不存在,则新建轨迹链,并记录源 IP 和源端口信息,转入步骤1,继续抓取下一个 DNS 查询数据包进行处理,否则,提取上一历史 TransID 记录的特征字节,与当前特征字节进行对比。如果前 $i-1$ 位均相同,第 i 位不同,则可以判断第 i 位的模式;若历史记录的第 i 位为0,当前为1,则正序,否则为反序。判断出第 i 位模式后与轨迹链中记录的第 i 位模式对比,若轨迹链中记录的第 i 位模式还未定,则将当前判定模式写入记录;若轨迹

链中记录与当前判定模式相同,则转入步骤1,继续获取下一个DNS查询数据包,若不同,则将轨迹链中记录模式重置为未定,然后再转入步骤1。

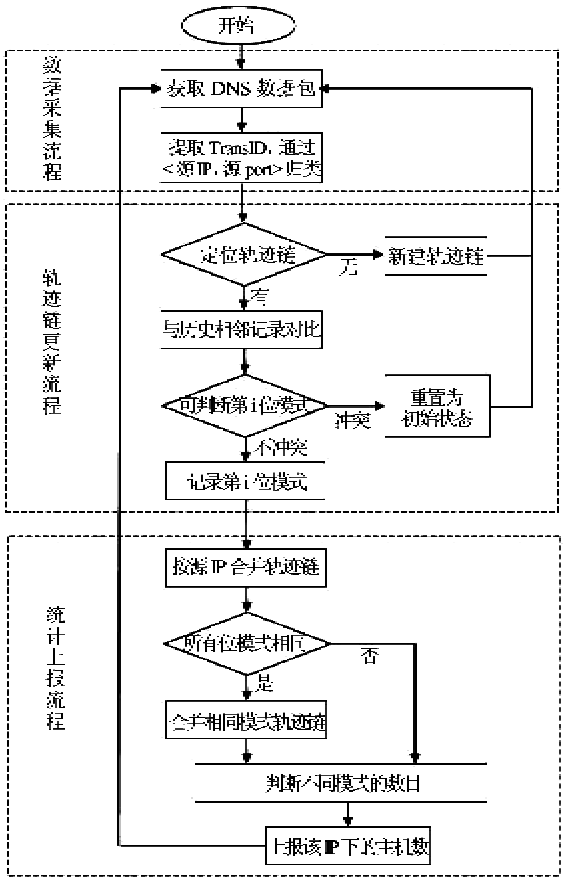

图1　利用DNS TransID轨迹检测共享接入主机流程

步骤3:统计上报流程。更新轨迹链之后,对同一源IP下的不同轨迹链进行合并,若两个不同源端口的轨迹链所有位模式均已确定,并且模式均相同,则两个轨迹链归并为一个,并且记录同属于该轨迹链的源端口信息。执行完合并之后,判断不同模式的数目,只要有一位模式不同,两个轨迹链的模式就必然不同,即使可能并非所有位模式均已确定。不同轨迹链模式的数目即检测到的主机数目,进行上报。然后转入步骤1,抓取下一个DNS查询数据包进行相同的处理。

至此,可以实时统计更新当前时刻检测出的共享接入的主机数目。

3.4　算法分析

算法提供了一种针对二值异或群循环轨迹识别的操作性很强的方案。如果通过求解相异或的元素值来区别不同轨迹之间的差别,则需要对所有元素异或结果对比匹配来确定,并且跳变情况的出现极大地干扰结果的判断,误差较大。而通过将二值异或群循环轨迹识别的问题转化为判断每一位的变化模式极大地降低了操作的复杂性,任何两个相邻的TransID变化值进行对比即能得出一位的变化模式,并且不需要完全得出所有位的变化规律即能进行判断,相当于只需要知道相异或元素值的一部分信息就能达到检测效果,检测的周期变短。此外,对异常情况和跳变情况容忍度增大,极大地降低了误判的几率,如果通过 p 位特征位进行检测,则最多可以检测 2^{p-1} 台主机,如果发生误判,则至少需要跳变 2^p-1 个TransID值才会发生。如 p 取4时,需要跳变15个值才能发生误判,p 取8时,需要跳变255个值才能发生误判,因此在正常跳变情况下,误判基本不会发生,算法准确度很高。

4　实验结果

实验中,通过在具备NAT功能的路由器下部署4台主机上网,在路由器出口处采集DNS查询数据包并执行算法。NAT后4台主机采用Windows XP SP2系统,其中dnsapi.dll和dnsrslvr.dll的版本号为5.1.2600.2180。在四台主机上所做的操作主要就是查看网页,当缓存中没有相关记录时,就会产生DNS查询请求。数据重传已经进行了相应处理。

图2显示了一台主机的DNS TransID随时间顺序变化的实际情况,特征位取4位,该主机的TransID轨迹实际上就是 $T(Z_2^4)\oplus 0xd$ 的二元异或群循环轨迹。不难看出,中间会出现一些跳变的情况,并且源端口不只有一个,SP1209表示对应数据包源端口是1209,SP1213表示对应数据包源端口是1213,这些源端口下的TransID值共同组成了该台主机的TransID轨迹,最终对该台主机模式的判断正是对同一源IP下的不同源端口的TransID变化模式的合并完成的。

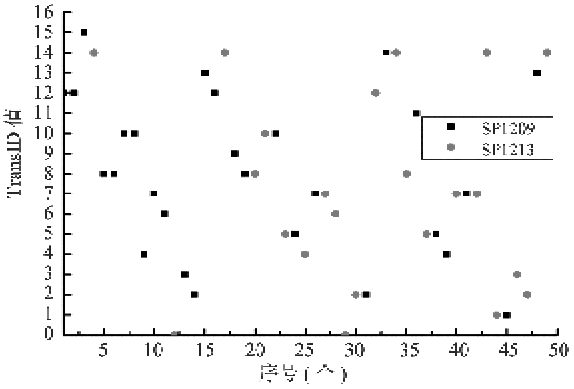

图2　某台主机DNS TransID序列变化情况

图 3 显示了不同特征位情况下确定轨迹模式所需要的最少的 TransID 个数情况。其中 Th 表示理论状态下所需 TransID 个数的最小值;Re _ NP 表示实验环境下在不考虑端口的情况下测得的判断所有位需要的 TransID 个数情况;Re _ P 表示实验环境下 NAT 公网 IP 下某端口所有位模式判断出来所需要的所有发出查询的 TransID 个数。从图中可以看出,实验环境不考虑端口时判断所有位所需 TransID 个数甚至比理论值都小,这是因为出现跳变的原因,某些会重复判断某位模式的 TransID 未出现,因此所有位判断使用的 TransID 数有所减少。整体上所需 TransID 个数与理论最小值相差不大。实际识别出不同轨迹所需的 TransID 个数还会更小,因为得出结论不一定需要所有位都判断出来。

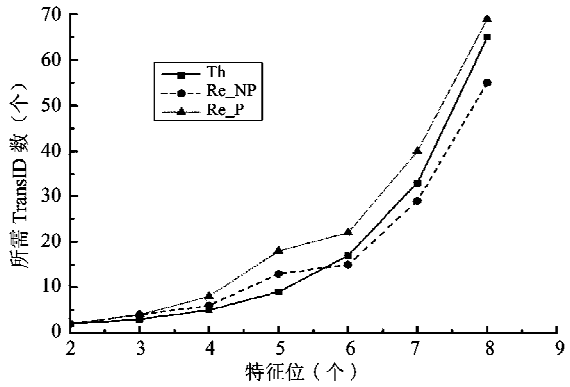

图 3 不同特征位情况下轨迹模式确定所需 TransID 个数

此外,实验中通过使用 4 位特征位判断 4 台主机个数结果非常准确,未发生误判情况。

5 结 论

多轨迹识别是业务监控识别需要解决的关键技术,针对不同轨迹提出快速高效的算法是重点研究的一个方向。本文首先对多轨迹识别问题建立了数学模型,定义了一类轨迹:二值异或群循环轨迹,并针对这类轨迹提出了高效的检测算法,通过将求解与基轨迹相异或的元素值的问题转化为求解每一特征位的模式问题,极大地提高了算法的可操作性和准确高效性。最后发现主机 DNS TransID 具备二值异或群循环轨迹特征,可利用该字段检测共享接入主机数目,通过实验表明,该算法需要较少的样本数据,所需数据与理论最小值相差极小甚至小于最小值,同时所需检测主机数越多,需要的样本数据也越多,对异常情况具有较强的容忍度,准确度很高,更好地适应检测共享接入主机数场景。下一步的工作将继续研究适用于共享接入检测的主机特征,并研究其轨迹模式,深化多轨迹识别算法。

参考文献

[1] Soldani C. Peer-to-peer behaviour detection by TCP flows analysis: [Engineer Dissertation]. Liege: University of Liege, France. 2004.5

[2] 韦安明, 王洪波, 程时端等. 高速网络中 P2P 流量检测及控制方法. 北京邮电大学学报, 2007, 30(005): 117-120

[3] Bellovin S M. A technique for counting NATted hosts. In: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement. New York: ACM, 2002.267-272

[4] Postel J. Internet Protocol. RFC 791,1981

[5] Stevens W R. TCP/IP Illustrated (vol. 1): the Protocols. Boston: Addison-Wesley Longman Publishing Co., 1993. 24-27

[6] Kohno T, Broido A, Claffy K C. Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2005, 2(2): 93-108

[7] Beverly R. A robust classifier for passive TCP/IP fingerprinting. *Lecture Notes in Computer Science*. 2004,3015: 158-167

[8] Li Z, Yuan R, Guan X. Accurate classification of the Internet traffic based on the SVM method. In: Proceedings of the IEEE International Conference on Communications, Glasgow, UK, 2007.1373-1378

[9] Dornhoff L L, Hohn F E. Applied Modern Algebra. New York: MacMillan Publishing Co., 1978.201-202

[10] 丁文霞, 卢焕章, 谢剑斌. 混沌二值序列对异或运算构成群的理论和实验证明. 系统工程与电子技术, 2006, 28(009): 1420-1422

[11] Srisuresh P, Egevang K. Traditional IP network address translator (Traditional NAT). RFC 3022, 2001

[12] Mockapetris P. RFC 1034: Domain names-concepts and facilities. http://www. ietf. org/rfc/rfc1034. txt: RFC,1987

[13] Mockapetris P. RFC 1035: Domain names-implementation and specification. http://www. ietf. org/rfc/rfc1035. txt: RFC, 1987

[14] Klein A. Microsoft Windows DNS stub resolver cache poisoning. http://www. trusteer. com/files/Microsoft _ Windows _ resolver _ DNS _ cache _ poisoning.pdf:Microsoft, 2007

[15] Microsoft Security Bulletin MS08-020-Important; Vulnerability in DNS client could allow spoofing. http://www. microsoft. com/technet/security/Bulletin/MS08-020. mspx: Microsoft, 2008

A multi-track separating algorithm based on binary XOR group

Zhu Hongliang[*,**], Li Rui[*,**], Yang Xin[*,**,***], Yang Yixian[*,**], Xu Guoai[*,**]
([*] Key Laboratory of Network and Information Attack & Defence Technology of MOE,
Beijing University of Posts and Telecommunications, Beijing 100876)
([**] Information Security Center, State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, Beijing 100876)
([***] Beijing Safe-Code Technology Co. Ltd., Beijing 100876)

Abstract

For there are no effective separating algorithms at present for binary XOR group circular track sets, the authors conducted the research on a method for rapid and efficient multi-track separation. First, a mathematical model of the multi-track separating theory was built, and a new type of track called the binary XOR group circular track was defined. Then, a new separating algorithm for the kind of track was proposed, by which the problem of solving the element that performs XOR operation with the base track can be converted into deciding the eigen bit mode of the track. It proved to be correct in theory and improved the operability greatly. Finally, the algorithm was applied to the shared-access host number detection to prove its effectiveness. The results indicate that this approach is with the high accuracy and tolerance of mis-report and can be widely used in network monitoring and the information security field.

Key words: multi-track separating, binary XOR group, eigen bit, shared access, network monitoring