基于三元符的公共模式的恶意 URL 检测

熊翠文[1,2,3]，张鹏[1,2]*，刘庆云[1,2]，谭建龙[1,2]

(1. 中国科学院信息工程研究所，北京 100093； 2. 信息内容安全技术国家工程实验室，北京 100093；

3. 中国科学院大学， 北京 100049)

摘　要：本文提出一种使用基于三元符的 URL 公共模式集检测恶意 URL 的方法，该方法首先将 URL 公共模式提取转换成 URL 的域名、路径名和文件名三个段的公共模式的拼接，然后通过以三元符为词项的动态倒排索引加快每个段的公共模式的计算，最后使用基于倒排索引检索的 URL 公共模式匹配被检测 URL，以判定其是否是恶意的。而且，该方法支持基于 Jaccard 的随机域名检测技术来检测包含随机域名的恶意 URL。大量的实验表明本方法具有较好的性能和扩展性。

关键词：恶意 URL 检测，公共模式，三元符，倒排索引

中图分类号：TP302　　**文献标识码**：A　　**文章编号**：

TCP: Trigrams-Based Common Patterns for Malicious URL Detection

XIONG Cui-wen[1,2,3], ZHANG Peng[1,2], LIU Qing-yun[1,2], TAN Jian-long[1,2]

(1. Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China;

2. National Engineering Laboratory for Information Security Technologies, Beijing 100093, China;

3. University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: This paper proposed an approach of malicious URL detection using trigrams-based common pattern of URL named TCP, where the common patterns were composed of common patterns of three segments of URL, namely domain segment, path segment and file segment. An inverted index based on trigrams was used to improve common pattern extraction of each segment. TCP used the common patterns based on inverted index to match with the detected URL, judged the URL is malicious or not. Moreover, TCP implemented a random domain name identification based on Jaccard to detect the malicious URLs with random domain name. Extensive experiments showed the TCP is efficient and scalable.

Key words: Malicious URL Detection, Common Pattern, Trigram, Inverted Index

1　引言

随着互联网技术发展，网络犯罪手段层出不穷，促使网络犯罪的网络威胁形式越来越多，导致识别网络威胁的难度大大增加。Ponemon Institute 发布的研究报告[1]表明，网络犯罪在 2013 年所造成的经济影响较过去 4 年增加了 78%，企业或组织平均每周遭受到 122 次成功的网络攻击，平均为每次网络攻击花费的成本超过了一百万美元。尽管网络防御水平在不断提高，但是网络犯罪集团也在不断增强其适应力，因此需要更好的方法来制止网络犯罪的发生。Wiki 定义网络威胁的一个主要特点是虽然使用其他协议或组件访问网络，但都使用 http 或 https 协议进行传输[2]，所以采用检测 URL 来判定网络威胁例如钓鱼网站等是可行的，Le 等人的研究[3]也表明使用 URL 字符串检测恶意 URL 是可行的。

基金项目：国家自然科学基金(61402464)；中国博士后基金(2013M541076)

Foundation Items: National Natural Science Foundation of China(61402464); China Postdoctoral Science Foundation of China(2013M541076)

*通讯作者 张鹏(pengzhang@iie.ac.cn)

然而，恶意 URL 为了减少被检测到的可能，可能会采用各种手段来隐藏自己。例如，Porras 等人[4,5]使用当前日期和时间作为种子每小时随机生成 250 到 50,000 个域名，使带有随机域名的 URL 难以被检测。其次，恶意 URL 与良性 URL 之间较高的相似度也很容易误导用户，例如将 login 篡改为 log1n 或将 index 篡改为 lndex 等，用户很可能误入这些网址导致信息泄露。因此要避免用户信息泄露，这些恶意 URL 需要在用户点击之前被检测出来。所以，好的恶意 URL 检测方法应能达到以下三个要求：

一、实时性，检测方法应能在短时间内检测出恶意 URL。在用户将要访问一个恶意 URL 时需要请求服务器，检测方法应能在恶意网页返回给用户之前提示用户该 URL 具有不良目的，并将恶意网页内容阻止在客户端外；

二、扩展性，检测方法应能有效地检测出新的恶意 URL。攻击者为了躲避正常的检测，会使用算法来生成随机域名增加检测的难度，检测方法应该能够检测出那些新的及不常见的恶意 URL；

三、有效性，检测方法应具有较高的精度。目前恶意 URL 在网络上的存在数量远不及良性 URL 的数量，要能精确地检测出恶意 URL 具有一定的挑战；另外，有的恶意 URL 只篡改了良性 URL 的一些关键词，给恶意 URL 和良性 URL 的区分带来了一定的挑战。

为此，本文提出的使用基于三元符的公共模式的恶意 URL 检测方法——TCP 方法，该方法由于只使用 URL 的词汇特征，不需要额外特征，节省很多时间开销。其中，公共模式直接从 URL 串中提取。根据 URL 的标准规范[6]，URL 字符串只包含字母数字字符和一些特定的符号例如“/”、“?”、“.”、“=”、“-”、“_”等，所以公共模式的提取过程只是对字符串进行处理，在目前的条件下，可以达到每秒百万量级。不仅如此，TCP 通过动态的倒排索引来检索可能存在公共模式的 URL，避免了不存在公共模式的 URL 对之间公共模式的计算，加快了整个 URL 数据集的公共模式计算，同时，TCP 使用有限自动机[7]来完成公共模式与被检测的 URL 匹配，并使用三元符检索那些可能匹配的公共模式，提高了公共模式匹配 URL 的效率。

文章的组织如下：第 2 节主要介绍目前的恶意 URL 检测方法的进展，并指出 TCP 方法与其不同之处，指明 TCP 方法的特点；第 3 节将介绍 TCP 方法中使用的概念及定义，以及 TCP 检测恶意 URL 的工作流程；第 4 节通过与经典的恶意 URL 检测方法的实验对比，验证 TCP 检测恶意 URL 的实时性、可扩展性和有效性；第 5 节是总结和工作展望。

2 相关工作

恶意 URL 检测方法大致可以分为三类，基于黑名单的方法、基于网页内容的方法和基于 URL 的方法。

第一类是基于黑名单的方法，例如 Google Safe Browser[8]、Netcraft Tool Bar、eBay Tool Bar 等的浏览器的黑名单机制。Zhang 等人在[9]中详细介绍了黑名单的工作原理，主要是通过比较用户访问的 URL 与黑名单中的 URL，若 URL 出现在黑名单中，则该 URL 为恶意的，否则为良性的。这种方法简单且准确度很高，需要通过人工标记、蜜罐、用户反馈、爬虫等方法来维护黑名单。然而，黑名单方法只能检测已出现过的恶意 URL，对于新的和包含随机域名的恶意 URL 无法检测。在这一点上，TCP 的特点是不需要被检测 URL 与训练集中的 URL 完全相同才认为是恶意的，也不需要维护大量的黑名单的 URL 列表，只需要通过倒排索引维护 URL 的公共模式集。被检测 URL 与 URL 公共模式进行匹配，然后比较两类公共模式匹配数以判断 URL 的恶意与良性。

第二类是基于网页内容的方法，由于恶意 URL 的网页内容具有某种特殊的目的或意义，因此网页内容也为恶意 URL 检测提供了丰富的特征。例如，Provos 等人[10]使用 URL 的网页内容特征检测恶意 URL，例如某些 javascript 是否出现，iFrames 是否越界。Moshchuk 等人[11]为检测恶意 URL 使用反间谍工具来分析下载的木马可执行文件。Zhang 等人[12]利用信息检索中的 TF-IDF 算法，分析网页中每个术语的 TF-IDF 值，借助鲁棒超级链接中词汇标签概念，在 Google 搜索引擎检索词汇标签，确定被检测网页的合法性。通过这种方法判断恶意 URL，首先需要获取网页的内容，然后对网页内容进行分析，这样会带来显著延迟，不适合高速的在线检测。在这一点上，TCP 的特点是不需要访问网页的内容，只需要访问 URL 字符串，更适合在线检测。

第三类是基于 URL 的方法，目前已有的方法

基本都是通过提取 URL 中的特征，例如 URL 的长度信息、服务器地理位置信息、服务器 IP 信息等，训练分类器以对 URL 进行分类。例如，Garera 等人[13]分析钓鱼网站的 URL 结构，总结出 4 种类型的 URL 结构，通过特征集合选取过程，选取页面特征、域名特征、类型特征和词汇特征等 18 种特征，最后利用 Logistic 回归过滤器对 URL 进行分类。Ma 等人[14-16]分析可疑 URL 的词汇特征和主机属性，运用词袋模型获得成千上万的特征，在词汇特征中，既考虑主机名长度、URL 长度、URL 中点号数等信息，又考虑对于 URL 中主机和路径中每一个词汇符号信息，最后运用词袋模型建立二值特征。在主机属性中考虑 IP 地址属性、WHOIS 属性、域名属性和地理位置属性。在这一点上，TCP 也可以划分为基于 URL 的方法，但是 TCP 只考虑 URL 的词汇特征，减少了 URL 多种特征的提取和计算带来的开销，且 Le 等人[3]的研究表明基于 URL 词汇特征检测恶意 URL 的可行性。

本文提出的 TCP 具有以下特点：第一，只使用基于 URL 字符串提取的公共模式作为特征，用来匹配待检测的 URL，根据匹配数判断 URL 是良性的还是恶意的；第二，使用动态的倒排索引过滤存在公共模式的 URL，加快公共模式的计算和匹配速度，进而提高 TCP 对恶意 URL 的检测速度；不仅如此，通过基于 Jaccard 的随机域名检测技术在一定程度上检测出随机域名的恶意 URL。第三，实验表明，相同规模的公共模式和 URL 黑名单列表，公共模式匹配的 URL 数远多于黑名单匹配的 URL 数，TCP 在一定程度上具有较好的扩展性。

3 TCP 方法的原理

3.1 概念及定义

首先给出 TCP 方法中涉及的概念，根据 URL 的标准规范[6]，URL 字符串包含三个不同含义的段：域名、路径名和文件名，三个段因含义不同它们之间的相关性不大，所以将 URL 字符串分解成三个段，然后逐段考虑。为了简化运算，本文规定将 URL 中的字母数字字符和特定的符号例如“?”、“=”、“-”、“_”等都当作常规字符对待，特别地，字符“/”作为段连接符和路径名分隔符，“.”作为域名和文件名分隔符，提取公共模式时常规字符不区分考虑。另外，域名、路径名、文件名三个段由相同字符集组成，所以公共模式形式相同，分别用 s_d、s_p、s_f 表示域名、路径名、文件名的公共模式，段公共模式匹配段的规则相同。

定义 1(段公共模式) 段公共模式是常规字符串 $s=c_1\cdots c_\ell$，其中 ℓ 是公共模式的长度，$c_i(1\le i\le \ell)$ 是 URL 标准文件[6]中的常规字符，或 c_i 是通配符“*”，通配符能匹配任意长度的常规字符串，但不包含“/”和“.”。对于任意的 $i(1\le i<\ell)$，如果 $c_i=*$，那么 $c_{i+1}\neq *$。规定只包含通配符的公共模式是非法的。

规则 1(段公共模式匹配段) 对于段公共模式 $s=c_1\cdots c_\ell$ 和段 $u=c'_1\cdots c'_m$，如果存在一个函数 $f:[1,m]\to[1,\ell]$ 满足条件：1、$\ell\le m$；2、对于任意的 $j(1\le j<m)$，有 $f(j)\le f(j+1)$；3、对于任意的 $i(1\le i<\ell)$，如果 $c_{i+1}\neq *$，则存在唯一的 $j(1\le j<m)$，使得 $f(j)=i$ 和 $c'_j=c_i$，则称段公共模式匹配段。

URL 公共模式是基于三个段公共模式的，用 URL 公共模式检测 URL 是良性的还是恶意的同样也基于段公共模式。

定义 2(公共模式) URL 公共模式(简称公共模式)由对应的三个段公共模式通过字段连接符连接而成，表示为 $P=s_d/s_p/s_f$。

规则 2(公共模式匹配 URL) 若公共模式的每个段公共模式与被检测 URL 的对应段匹配，则公共模式匹配该 URL。

3.2 TCP 工作流程

本文使用的数据集是通过开源软件 Larbin[17]在某个时间段内从特定网站上抓取的 URL。TCP 从获取数据集到恶意 URL 检测结束的整个处理流程如图 1 所示，TCP 检测恶意 URL 包括数据集预处理、公共模式提取、URL 判定三个模块。

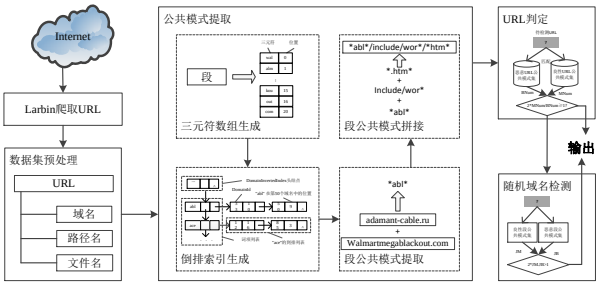

图 1 TCP 检测恶意 URL 工作流程

数据集预处理模块负责将 URL 分解成三个段；公共模式提取模块负责将段分割成三元符，建立倒

排索引，基于以三元符为词项的倒排索引提取每个段的公共模式，然后拼接对应段公共模式得到公共模式；URL 判定模块负责用公共模式匹配 URL，然后判定被检测 URL 的是否是恶意 URL。

TCP 获取 URL 数据集之后，首先经过数据集预处理模块将 URL 分解成域名、路径名、文件名三个段。然后通过公共模式提取模块对三个段做相同处理分别提取段公共模式，以域名为例进行说明。将域名分解成多个三元符，建立以三元符为词项的动态倒排索引。取三元符作为词项的原因是，任意两 URL 串之间具有相同二元符、四元符的概率分别为 95.7%和 33.6%，而具有相同三元符的概率为 75.8%[18]。所以，为避免提取的公共模式太短或提取太频繁，TCP 规定两个 URL 之间至少有一个相同的三元符才提取公共模式。基于动态倒排索引提取段公共模式，段公共模式满足 3.1 小节中的定义 1，然后将相应的段公共模式拼接得到公共模式。最后使用 URL 判定模块根据规则 1、2 用公共模式匹配 URL，判断被检测 URL 的性质，并输出结果。

3.2.1 数据集预处理

TCP 使用开源软件 Larbin[17]在特定时间内从特定网站上抓取 URL，获得恶意 URL 和良性 URL 两个数据集，分别对两类数据集进行处理。因为处理的 URL 采用的协议都是 http 或 https，所以不考虑协议，URL 分解之前先将“协议：//”从 URL 中分离。然后根据 URL 字符串的特点将所有的 URL 分解成三个段，即域名、路径名和文件名。例如 URL0 "walmartmegablackout.com/include/wordpress/login.htm"分解成：域名 Domain0 = "walmartmegablackout.com"、路径名 Path0 = "include/wordpress"、文件名 FileName0 = "login.htm"。分别将恶意的和良性的数据集中所有的域名、路径名和文件名组合起来得到三个段集——域名集、路径名集和文件名集。

3.2.2 公共模式提取

TCP 提取公共模式分为以下四步：

1)三元符数组生成：TCP 经过数据集预处理模块分别得到两类数据集的三个段集，前面已经提到构成域名、路径名、文件名三个段的字符集相同，从三个段集中提取段公共模式的方法相同。为了避免提取的段公共模式太短或提取操作太频繁，TCP 将段分解成多个三元符，然后以三元符作为词项动态构建倒排索引。域名分解成多个三元符的算法如算法 1 所示，其中 Count 用来记录域名中包含的三元符个数，其值不会超过域名中常规字符数减 2，TrigramArray 是用于记录域名的三元符及其位置的三元符结构体数组。算法根据域名中的“.”将域名分解成子串，每个子串分别分割成三元符，这样保证三元符中的字符来自同级域名，每个域名的三元符保存在一个三元符数组 TrigramArray 中。算法遍历一次域名就能将域名分解成三元符数组，时间复杂度是线性的。文件名分解成三元符的方法与域名一致，而路径名以“/”将不同文件夹分隔开，计算方法基本与域名分解方法相同。

Algorithm 1: SplitDomainIntoTrigrams

```
01: Input: Domain, Count
02: Output: an array of trigrams of Domain
03: initial TrigramArray← ø
04: CurrentPos = 0
05: Split Domain into SubStrings using "."
06:     while SubString != ø
07:         if strlen(SubString) ≥ 3
08:             Split SubString into successive Trigrams
09:             TrigramArray←Trigram and Position
10:         else
11:             SubString regard as Trigram
12:             TrigramArray←Trigram and Position
13:         end if
14:         Count++
15:     end while
16: return TrigramArray and write back Count
```

2)倒排索引生成：TCP 每得到一个三元符数组，基于三元符为每个段动态地建立倒排索引，域名的倒排索引 DomainInvertedIndex，如图 2 所示。倒排索引的词项链表的节点包含三个字段：Trigram、指向下一个 Trigram 的指针和指向包含该 Trigram 的倒排列表的指针。每个词项对应的倒排列表的节点包含该三元符出现的域名编号、在域名中的位置和指向下一个包含该 Trigram 的域名节点的指针，倒排列表按域名编号递增。同样的方法生成相同结构的路径名的倒排索引 PathInvertedIndex 与文件名的倒排索引 FileInvertedIndex。

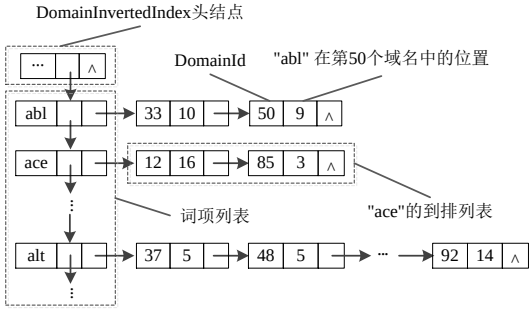

图 2　域名倒排索引

3)段公共模式提取：三个段公共模式提取方法相同，以域名为例说明。TCP 遍历域名的倒排索引 DomainInvertedIndex 查找可能存在段公共模式的域名，出现在同一倒排列表中的域名两两提取段公共模式。通过遍历两域名对应的三元符数组提取两域名的公共模式，具体过程如算法 2 所示，该算法对两域名只能提取一个段公共模式。据实验统计，数据集中两域名存在多于一个段公共模式的概率不超过 2%[18]，因此两域名之间只计算一个段公共模式是可行的。段公共模式的提取是根据两域名的三元符数组中的三元符在域名中的位置来计算的，根据三元符在域名中出现的位置不同来判定是否将三元符或三元符中的字符或通配符写入段公共模式。算法遍历两个段的三元符数组，时间复杂度不超过两数组长度之积，也是线性的。Domain0 与 "adamant-cable.ru"提取的段公共模式为"*abl*"，最后将所有的域名段公共模式存入 DomainPatternSet 中。用同样的方法处理路径名和文件名，得到路径名公共模式集 PathPatternSet 和文件名的公共模式集 FileNamePatternSet。

Algorithm 2: ExtractDomainCommonPattern

```
01: Input: DomainId1, DomainId2
02: Output: the common patter of Domain1 and Domain2
03: initial CommStr←ø
04: Traverse TrigramArrays of Domain1 and Domain2
05:     if TrigramArray1[i].Trigram ==
            TrigramArray2[j].Trigram
06:         if CommStr == ø
07:             Judge( Position1i, Position2j)
08:                 CommStr= Trigram | ".*"+Trigram | "*"+Trigram
09:         else
10:             Judge( Position1i, Position2j)
11:                 Append(CommStr, the last letter of Trigram |
                        "*."+Trigram | "*"+Trigram | Trigram)
12:         end if
13:         Position1 = Position1i; Position2 = Position2j;
14:     end if
15: if any position is not the last Trigram of two Domain
16:     CommStr append "*"
17: end if
18: return CommStr
```

4)段公共模式拼接：根据相同的 URLId 用段连接符"/"拼接对应的段公共模式，URL0 与 URL1 "adamant-cable.ru/include/world/index.html" 的公共模式为"*abl*/include/wor*/*.htm*"。最后所有的恶意 URL 公共模式输出到 URL 判定模块。同样地，良性 URL 公共模式也输出到 URL 判定模块。

3.2.3 URL 判定

使用 3.1 小节中的规则 1 和 2 用两类公共模式对被检测 URL 进行匹配。TCP 研究过程使用的数据集中，恶意 URL 的训练集与良性 URL 的训练集大小之比为 1:2，恶意 URL 公共模式集中的公共模式匹配被检测 URL 的个数记为 MNum，良性 URL 公共模式集中的公共模式匹配被检测 URL 的个数记为 BNum。数据集及提取的公共模式集的特点，恶意 URL 公共模式集与良性 URL 公共模式集之间基本无交集，即两类公共模式集很少有相同的公共模式，则这种数据集适合使用简单的线性分类器[19]，这里使用有限自动机进行判定。TCP 判定被检测 URL 是良性的还是恶意的通过比较 MNum 与 BNum，若 2*MNum/BNum≥1，则判定被检测 URL 为恶意的，否则判定为良性的。

3.3 随机域名检测

目前，很多恶意 URL 通过随机生成域名的方式来躲避域名检测[4,5]。为了进一步提高准确率，TCP 在 URL 判定中引入随机域名检测机制。在这方面，Yadav 等人[20]通过计算域名的一元符和二元符的 KL 距离、Jaccard 指数和编辑距离以判定域名是良性的还是恶意的，并且对于不同的场景使用不同的评价指标，得出 Jaccard 指数判定效果最佳的结论。因此，TCP 也引用 Jaccard 指数来处理随机域名的情况，鉴于提取公共模式时使用三元符作为词项，对于随机域名的衡量继续使用三元符作为基。对于不能用有限自动机判定的恶意的 URL，通过使用如下的 Jaccard 公式计算：

$$JM = \frac{A \cap B}{A \cup B}$$

其中，A 表示被检测 URL 的域名三元符数组中三元符个数，B 表示恶意 URL 的域名段公共模式集中所有不重复的三元符个数，$A \cap B$ 表示 A 与 B 之间较小的值，而 $A \cup B$ 表示 A 与 B 之间较大的值。同样的方法计算被检测 URL 与良性 URL 的域名段公共模式集的 Jaccard 指数 JB，比较 JM 与 JB 之间的大小若 $2*JM/JB \geq 1$，则该 URL 是恶意的。

4 实验与分析

实验使用的两类 URL 数据集——恶意 URL 数据集和良性 URL 数据集都来自网络上的公开数据集，它们是通过使用开源软件 Larbin[17]在 2014 年 10 月 13 日同一天从两类网站上抓取并去重的。恶意 URL 数据集由从 Phish Tank[21]和 Malware Patrol[22]网站上爬取 4,000,000 个恶意 URL 组成，

良性 URL 数据集由从 Google 和 DMOZ 网站上爬取 8,000,000 个良性 URL 组成。测试集的来源与获取方法与训练集相同，大小为 2,000,000 的测试集中包含 800,000 个恶意 URL 和 1,200,000 个良性 URL。实验使用的计算机拥有 4GB 的内存和 3.2GHz 的 CPU。每次实验中，URL 被随机抽取，训练集中恶意 URL 与良性 URL 的比例为 1:2，测试集中恶意 URL 与良性 URL 的比例为 2:3，每次实验训练集与测试集的比例为 5:1。

为验证 TCP 方法的有效性，本文运行 Ma 等人[15]提出的 CW 算法的代码，相同环境下基于相同数据集运行 CW 算法和 TCP 方法，它们的误判数和漏判数分别如图 3 和 4 所示，误判数是指将恶意 URL 判定为良性 URL 的数量，漏判数是指将良性 URL 判定为恶意 URL 的数量。图 3 和 4 中横坐标表示训练集中 URL 的总数，单位为十万个，纵坐标表示误判数和漏判数，单位为百个。实验数据表明，TCP 方法的漏判数和误判数都较 CW 低，说明 TCP 方法对于检测恶意 URL 是有效的。

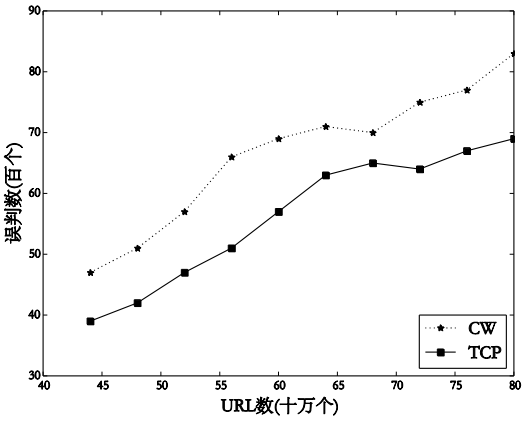

图 3　TCP 与 CW 误判数比较

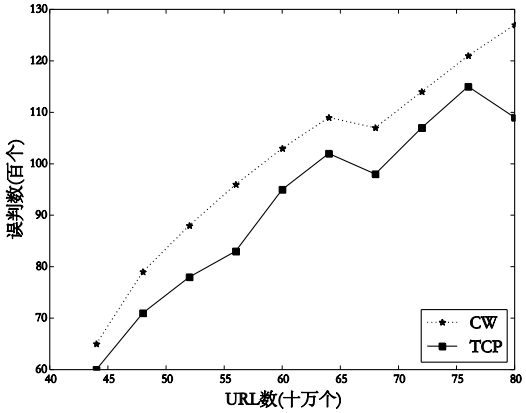

图 4　TCP 与 CW 漏判数比较

为验证 TCP 方法的检测速度，图 5 比较了在相同环境和相同数据集下运行 CW 算法和 TCP 方法检测恶意 URL 的时间开销，图中横坐标表示训练集中 URL 的总数，单位为十万个，纵坐标表示从开始运行两种方法到检测结束所使用的时间，单位为秒。实验数据表明 TCP 在相同数据集下检测恶意 URL 的时间少于 CW 算法，且增长速度也小于 CW 算法，说明 TCP 方法的实时性较好。这是因为 TCP 采用动态倒排索引避免了冗余的计算，公共模式提取算法时间复杂度时线性的，且将公共模式按三元符在 ASCII 表中的顺序有序排列，匹配之前先根据倒排索引查找可能匹配的公共模式，减少冗余的匹配，所以 TCP 方法检测恶意 URL 速度很快。

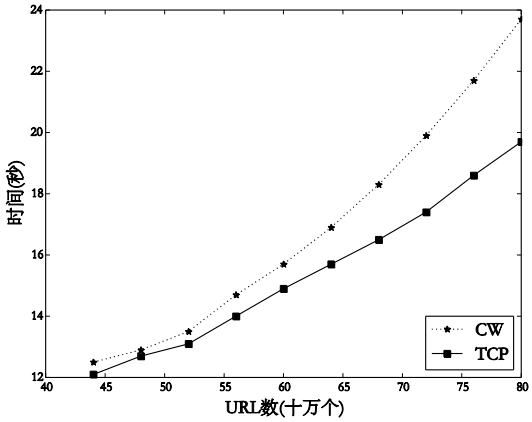

图 5　TCP 与 CW 运行时间比较

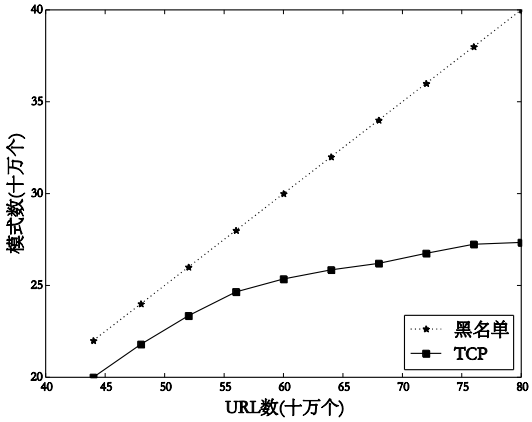

图 6　TCP 与黑名单模式数比较

为验证 TCP 方法的扩展性，图 6 对比了检测相同数量的恶意 URL 时，TCP 所使用的公共模式数量和黑名单所使用的 URL 的数量，横坐标表示检测的恶意 URL 的个数，单位为十万个，纵坐标表

示模式数，单位是十万个，这里将黑名单中的 URL 也看作是模式。从图中可以看出，公共模式的数量近似对数增长，而黑名单中 URL 数量呈现线性增长，这表明 TCP 的扩展性较好，这是因为 TCP 的一个公共模式能够匹配多个 URL，在一定程度上，能够检测出恶意的 URL。

5 结论

本文针对恶意 URL 检测提出一种使用基于三元符的公共模式的恶意 URL 检测方法——TCP 方法，该方法通过三元符为词项的公共模式提取以及动态倒排索引提高了恶意 URL 检测效率。对于随机域名产生的恶意 URL，该方法通过基于 Jaccard 的随机域名检测技术进行进一步判定。在相同实验环境下，通过与经典 CW 方法和黑名单方法的大量比较实验表明，TCP 方法在检测恶意 URL 时具有较好的实时性、扩展性和有效性。然而，TCP 还存在一些问题。例如，只通过简单的 Jaccard 指数检测包含随机域名的恶意 URL 的方式还不够完善。因此，在下一步工作中，我们打算根据域名与 IP 地址之间的映射关系，检测包含随机域名的恶意 URL。另外，目前的实验环境是在每秒 5000 个 URL 的小流量进行的在线测试，下一步将在国家骨干网的大流量环境下测试 TCP 方法的效果。

参考文献

[1] http://www.hpenterprisesecurity.com/ponemon-2013-cost-of-cyber-crime-study-reports[EB/OL]. 2015.

[2] http://en.wikipedia.org/wiki/Web_threat[EB/OL]. 2015.

[3] LE A, MARKOPOULOU A, FALOUTSOS M. Phishdef: url names say it all[A]. the 30th IEEE Internation Conference on Computer Communication[C].Shanghai, China. 2011.

[4] PORRAS P, SAIDI H. YEGNESWARAN V. Conficker C P2P Protocol and Implementation[R]. SRI International Tech. 2009

[5] PORRAS P, SAIDI H. YEGNESWARAN V. An Analysis of Conficker's Logic and Rendezvous Points[R] 2009.

[6] https://url.spec.whatwg.org/[EB/OL]. 2015.

[7] http://en.wikipedia.org/wiki/Automaton[EB/OL]. 2015.

[8] LIKARISH P, JUNG E. Leveraging Google SafeBrowsing to Characterize Web-based Attacks[A]. Association for Computing Machinery[C]. Chicago, IL, USA. 2009.

[9] ZHANG J, PORRAS P, ULLRICH J. Highly Predictive Blacklisting[A]. Proceedings of the 17th conference on Security symposium[C]. San Jose CA. 2008.

[10] PROVOS N, MAVROMMATIC P, RAJAB M A, et al. All Your Iframes Point to Us[A]. 17th Usenix security symposium[C]. San Jose CA. 2008.

[11] MOSHCHUK A, BRAGIN T, GRIBBLE S D, et al. A Crawler-base Study of Spyware on the Web[A]. Network and Distributed System Security Symposium[C]. Geneva, Switzerland. 2006.

[12] ZHANG Y, HONG J, CRANOR L. CANTINA: A Content-Based Approach to Detecting Phishing Web Sites[A]. 16th International World Wide Web Conference[C]. Banff, Alberta, Canada. 2007.

[13] GARERA S, PROVOS N, CHEW M. A Framework for Detection and Measurement of Phishing Attacks[A]. The 5th ACM Workshop on Recurring Malcode[C]. Alexandria, Virginia, USA. 2007.

[14] MA J, SAUL L K, SAVAGE S, et al. Beyond Blacklists: Learning to Detection Malicious Web Sites from Suspicious URLs[A]. the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining[C]. Paris, France. 2009.

[15] MA J, SAUL L K, SAVAGE S, et al. Identifying Suspicious URLs: an Application of Large-scale Online Learning[A]. In proceedings of the 26th International Conference on Machine Learning[C]. Montreal, Canada. 2009.

[16] THOMAS K, GRIER C, MA J, et al. Design and Evaluation of a Real-Time URL Spam Filtering Service[A]. Proceedings of the 2011 IEEE Symposium on Security and Privacy[A]. San Francisco, CA. 2011.

[17] http://larbin.sourceforge.net/index-eng.htm[EB/OL]l. 2014.

[18] HUANG D, XU K, PEI J. Malicious URL Detection by Dynamically Mining Patterns Without Pre-defined Elements. The 22nd World Wide Web Conference. Rio de Janeiro, Brazil. 2013.

[19] 韩家炜等. 数据挖掘——概念与技术[M]. 北京：机械工业出版社. 2012.
HAN J W, et al. Data Mining Concepts and Techniques[M]. Beijing: China Machine Press. 2012.

[20] YADAV S, REDDY A K, RANJAN S. Detecting Algorithmically Generated Malicious Domain names[A]. International Micrographics Congress[C]. Melbourne, Australia. 2010.

[21] http://en.wikipedia.org/wiki/Phishtank [EB/OL]. 2014.

[22] http://www.malware.com.br/[EB/OL]. 2014.