基于扩展前缀树的协议格式推断方法

洪 征,田益凡,张洪泽,吴礼发
HONG Zheng, TIAN Yifan, ZHANG Hongze, WU Lifa

解放军陆军工程大学 指挥控制工程学院,南京 210000
Institute of Command and Control Engineering, Army Engineering University of PLA, Nanjing 210000, China

HONG Zheng, TIAN Yifan, ZHANG Hongze, et al. Extended prefix tree based protocol format inference. Computer Engineering and Applications, 2018, 54(12):14-20.

Abstract: Network protocol format inference is of great significance in many network security applications. Most existing protocol format inference methods suffer from high computation complexity and low accuracy. A extended prefix tree based protocol format inference method is proposed in the paper. Firstly, the candidate keywords are obtained through *N*-gram word segmentation and merged into protocol keywords of different lengths according to mutual information. On the basis of protocol keywords, the extended prefix tree is constructed according to protocol keyword sequences, and the initial clustering is performed on the extended tree. Then, through segmental multiple sequence alignment based on the extended prefix tree, the similar format will be combined and the precise protocol format can be obtained. Compared with traditional format inference methods, the proposed method reduces the time complexity of inference. Experimental results show that the proposed method performs well for both text protocols and binary protocols.

Key words: protocol format inference; mutual information; extended prefix tree; multiple sequence alignment algorithm

摘 要:对未知网络协议进行协议格式推断在网络安全领域具有重要意义。现有的协议格式推断方法存在时间复杂度高、精确度较低等问题。提出了一种基于扩展前缀树协议格式推断方法。该方法首先通过*N*-gram分词获取候选协议关键词,使用互信息进行合并得到不同长度的协议关键词。在此基础上,依据与报文相对应的关键词序列构建扩展前缀树,实现对报文样本的初步聚类。而后,在扩展前缀树的基础上采用分段的多序列比对方法获取精确的协议格式。实验结果表明,该协议格式推断方法对于文本协议和二进制协议都能够取得理想的推断效果。

关键词:协议格式推断;互信息;扩展前缀树;多序列比对算法
文献标志码:A **中图分类号:**TP393.08 doi:10.3778/j.issn.1002-8331.1803-0494

1 引言

协议规范是对网络协议语法、语义以及同步等信息的具体描述,在网络安全领域扮演着重要角色。僵尸网络中,攻击者使用C&C(Command and Control)协议来控制存在漏洞的主机实施DDoS(Distributed Denial of Service)攻击等恶意行为,网络管理员需要依据C&C协议规范来发现和分析僵尸网络[1]。入侵检测系统中,需要基于协议规范从繁杂的网络流量中辨识出恶意流量。在模糊测试过程中,可以利用协议规范指导测试用例生成以实现更加高效的自动化漏洞挖掘[2-3]。然而出于利益、版权、安全等因素,软件厂商和个人往往不愿意公开协议细节,如微软使用的网络文件共享SMB(Server Message Block)协议,Oracle数据库访问的TNS(Transparence Network Substrate)协议以及微信、QQ等即时通信软件所使用的协议都没有公开协议细节,这些未知协议在网络中的广泛使用,给安全防护带来了极大的阻碍。

对于未知协议而言,目前主要通过协议逆向分析方法[4]获取其协议规范。依据分析对象的不同,逆向分析方法可分为两类:基于网络流量的分析方法和基于指令执行轨迹的分析方法。基于网络流量的分析方法对截获的网络数据流进行分析,通过生物信息学、统计分析、

基金项目:国家重点研发计划(No.2017YFB0802900)。
作者简介:洪征(1979—),男,博士,副教授,主要研究方向:网络空间安全,E-mail:hz5215@163.com;田益凡(1992—),男,硕士研究生,主要研究方向:网络空间安全。
收稿日期:2018-03-30 **修回日期:**2018-05-15 **文章编号:**1002-8331(2018)12-0014-07

数据挖掘等方法，对报文样本进行聚类分析，依据相同格式报文在取值上的相似性，分析获取协议语法、语义信息。基于指令执行轨迹的分析方法以协议解析过程中的指令执行轨迹为分析对象，将协议输入数据作为污点数据源，利用动态污点分析(Dynamic Taint Analysis)方法[5]跟踪数据解析过程，依据协议解析程序如何使用污点数据以及相应的上下文信息获取协议规范。

基于指令执行轨迹的分析方法通常具有较高的准确率，能够获得更全面的语义信息，但这类分析方法的实施需要分析人员拥有协议终端的可执行程序，并要求依据协议解析环境进行分析，依赖于底层平台，无法移植，通用性不强。基于网络流量的分析方法以网络数据流为输入，虽然其准确率依赖于捕获样本的丰富程度，但实现灵活，适用于各种应用场景，能够实施自动化分析。本文针对基于网络流量的分析方法进行研究，重点关注协议语法以及语义信息的提取。

目前国内外已有一些针对协议格式提取的研究成果，PI(Protocol Information)项目[6]是最早提出的基于网络流量的协议格式推断方法，引入了生物学中的多序列比对算法，通过将报文样本进行比对对齐的方式提取协议格式信息。Cui等人提出一种基于递归聚类的协议格式提取方案Discoverer[7]，按照文本和二进制两种属性对报文字节流进行标注，得到报文属性序列，并对其进行序列比对得到初始聚类结果，随后利用格式标志(Field Distinguish)字段进行递归聚类获取协议格式。Krueger提出一种基于统计的格式推断方法PRISMA[8]。该方法首先利用自然语言处理中的*N*-gram思想以及分隔符对报文序列进行分词得到候选特征词，进而使用统计学方法对候选特征词进行筛选构建特征矩阵，并以特征词为基础描述报文序列。Zhang等人则提出一种基于专家投票算法的特征词提取方法ProWord[9]，并将其应用于流量分类。Luo等人提出一种基于位置信息的协议格式推断方法AutoReEngine[10]，采用Apriori算法获取特征词，利用位置信息对特征词进行合并和筛选，得到协议关键词集合，最终再次利用Apriori算法对协议关键词进行组合提取协议格式及协议状态机。

以上方案都存在一些缺陷：PI项目直接对整个报文样本进行序列比对，时间复杂度较高。Discoverer虽然利用报文属性序列和格式标志字段降低了时间复杂度，但其采用常见文本类分隔符对报文样本进行划分的处理方法并不适用于二进制协议。此外，对于未知协议而言哪些分隔符会在协议中被使用具有很大的不确定性。PRISMA中对文本协议使用分隔符方法进行划分，也存在类似问题。且PRISMA在处理二进制协议时，采用*N*-gram方法选取固定长度的特征词，而实际报文中特征词的长度并不相同，固定长度的特征词的权重也会受到随机串的影响。AutoReEngine通过Apriori算法结合位置信息提取协议关键词，对于字段位置可变的协议，如HTTP、SIP协议，可能遗失部分关键词。

为了提高协议格式推断方法的通用性以及准确率，本文提出一种基于扩展前缀树的应用层未知协议格式推断方法，实现了原型系统EPT-PFI(Extended Prefix Tree based Protocol Format Inference)。EPT-PFI以截获的网络报文作为输入，通过*N*-gram方法对报文样本进行分词，结合点间互信息PMI(Point Mutual Information)对候选关键词进行合并提取协议关键词，在此基础上对报文样本进行标注获得协议关键词序列。而后构建扩展前缀树(Extended Prefix Tree，EPT)描述协议关键词序列，并基于扩展前缀树实现分段的多序列比对。最后，依据协议报文结构对扩展前缀树进行合并，获取协议格式信息。

2　基于扩展前缀树的协议格式推断

2.1　问题定义

协议规范由协议格式和协议状态机两部分构成。协议状态机指定了合法报文的传送顺序，通过特定的报文发送序列构成特定会话(Session)以实现所需的功能。协议格式指定了会话中每条报文的语法以及语义信息，只有符合协议格式的才是合法报文，才能被协议终端正确解析。协议语法规定了报文的字段、结构以及字段语义，定义了组成报文的每一个字段(Field)的关键词、数据类型、位置以及长度等具体信息。协议语义则定义了每一个字段在协议解析过程中所代表的实际意义，如HTTP协议(Hyper Text Transfer Protocol)请求报文中的“HOST”字段指定了请求资源所在的主机和端口号。常见的协议报文主要涉及以下几种形式[11]：

(1)二进制形式，即预先规定若干个固定比特位为一个字段，字段内容即代表该字段的取值，如DNS协议的前16个比特位标识DNS ID号。

(2)ASCII码形式，即采用易于理解的ASCII字符表示字段含义，使用预定义的分隔符(空格、回车换行等)作为字段结束标志，如MIME协议报文中的协议版本字段“MIME-Version”。

(3)TLV(Type-Length-Value)形式，即使用固定长度字节表示字段类型，固定长度字节的Length表名该字段取值的字节数，而其后紧跟的指定长度内容即字段取值，如eDonkey协议格式。

(4)指针形式，即采用指针指出某一个字段的开始和结束，如DNS协议响应报文中的回答计数指定了回答区域的条目数。

通信报文中一些字段是固定字段，一些是可变字段。但由于协议规范以及使用条件的制约，同种协议中同类格式的报文样本往往会体现出一些统计相似性或相关性，具体来说就是在报文样本中有一些具有固定模

式、频繁出现的比特串或字符串。文献[11]将这些具有固定模式的串统称为协议关键词。以图1所示HTTP协议报文为例,请求报文中的请求方法字段“GET”和协议版本字段“HTTP/1.1”,响应报文中的响应码“200 OK”,以及“Key-Value”格式的字段中起标识作用的字符串“Key”,如“Host”、“Connection”和“Date”等,这些能够标识报文类型或者协议字段的字符串都可以被称为协议关键词。

```
GET /assets2/images/ok.png HTTP/1.1
Host: my.csdn.net
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
Accept: image/webp,image/*,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: uuid_tt_dd=-3462333295814227752_20160704; lzstat_uv=15764486247733991041|3507483

HTTP/1.1 200 OK
Server: openresty
Date: Thu, 16 Mar 2017 00:14:13 GMT
Content-Type: image/png
Content-Length: 1340
Connection: keep-alive
Keep-Alive: timeout=20
```

图1　HTTP协报文示例

另外,由于分隔符通常在单个报文中多次出现且位置变化较大,对于协议分析会造成较大困扰,所以本文在沿用文献[11]定义的基础上,将分隔符排除在协议关键词之外。而分隔符通常为一到两个字节,如逗号、回车换行符等,可以通过设置*N*-gram分词方法中*N*的取值来过滤掉分隔符。

本文的协议格式提取方案建立在协议关键词分析的基础之上。现有研究采用分隔符对文本协议进行分词以得到协议关键词,但未知协议的分隔符存在较大不确定性。对于二进制协议,目前方法通常采用*N*-gram提取关键词,但所得到的关键词为固定长度*N*,而实际协议关键词的长度往往不固定,所以这类方法与实际存在出入。论文提出一种基于点间互信息的协议关键词提取方法,适用于二进制协议和文本协议。首先采用*N*-gram分词方法对报文样本进行处理,在得到固定长度的候选关键词后,结合点间互信息对候选关键词进行合并,推断获得不同长度的协议关键词。

在信息论中,熵是对事物不确定性的度量,而点间互信息(Point Mutual Information)是在熵的基础上提出来的概念,这一概念经常被用于自然语言处理以及数据挖掘领域来衡量两个单词间的相关程度。单词w_i,w_j之间的互信息可以定义为:

$$PMI(w_i,w_j)=\frac{p(w_i,w_j)}{p(w_i)p(w_j)}$$

$p(w_i,w_j)$表示单词w_i,w_j同时出现在一个报文中的概率,$p(w_i)$表示单词w_i出现在报文样本中的概率,$p(w_j)$表示单词w_j出现在报文样本中的概率。$PMI(w_i,w_j)$衡量的是两个单词之间的相关性,即确定了一个单词后另一个单词的不确定性(即熵值)的减少量,$PMI(w_i,w_j)$越大,则两个单词之间的相关性越高。

对于通过分词获得的候选关键词而言,如果两个候选词是一个协议关键词的子串,那么它们之间将具有较大的相关性。本文引入点间互信息来衡量候选关键词之间的相关性。如果两个候选词位置相邻、存在$N-1$个连续的相同字符(如“HTT”和“TTP”)且相关性较大,则将两者合并为一个长度大于N的候选词,如此重复以获得任意长度的协议关键词。

获取协议关键词后,依据报文样本对应的协议关键词序列构建扩展前缀树,实现报文样本的初始聚类以及同类报文的分段。随后,采用分段的多序列比对获取报文的详细格式信息。并通过对扩展前缀树进行合并以减少冗余,得到最终协议格式。

2.2　方法概述

本文提出的协议格式推断方法,主要包括4个处理阶段:预处理阶段、协议关键词提取阶段、报文结构提取阶段以及协议格式合并阶段,主要流程如图2所示。其中预处理阶段的主要工作是对原始数据流进行划分获取报文样本集合。协议关键词提取阶段将利用*N*-gram分词方法对报文样本进行处理,结合互信息获取协议关键词序列。报文结构推断阶段依据协议关键词序列构建扩展前缀树,通过基于扩展前缀树描述的分段多序列比对推断协议格式。协议格式合并阶段将基于报文结构对扩展前缀树进行合并,得到详细的协议格式信息。

2.2.1　预处理阶段

对于未知协议而言,分析对象是连续的网络数据流,由于网络分层结构以及分片机制,往往需要以会话

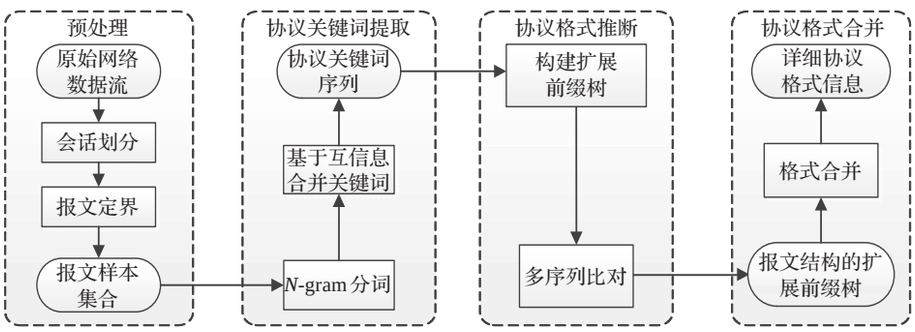

图2　原型系统EPT-PFI的主要处理流程

和报文为粒度对数据流进行分割得到便于分析的报文样本集合[12]，以会话为粒度的分割被称为会话划分，以报文为粒度的分割被称为报文定界。

会话划分是从连续的网络数据流中分离出通信实体间的单次会话，主要依靠底层协议提供的信息实现。对于应用层协议而言，可以使用协议五元组<源IP地址，源端口号，目的IP地址，目的端口号，传输层协议类型>得到独立的通信会话。

报文定界是从一次独立会话中分离出单个协议报文。对于应用层协议而言，可以利用传输层协议的报文首部和首部的长度字段实现。定界之后得到的报文样本作为后续阶段的输入。

2.2.2　协议关键词提取阶段

此阶段的目的是从经过预处理的报文样本中提取协议关键词，并使用协议关键词对报文进行标注得到协议关键词序列。原始报文样本集合可以表示为 $S=\{m_1, m_2, \cdots, m_i, \cdots, m_s\}$，其中 m_i 为样本集合中的某一个报文，$m_i=b_1^i b_2^i \cdots b_j^i \cdots b_M^i$，$b_j^i$ 表示报文 m_i 中的第 j 个字符。得到初始报文样本后，采用*N*-gram分词方法对报文样本进行逐一处理，得到所有在样本集合中出现过的长度为 N 的字符串，如对报文 $m_i=b_1^i b_2^i \cdots b_j^i \cdots b_M^i$ 进行*N*-gram分词，将得到 $w_1=b_1^i b_2^i \cdots b_N^i, \cdots, w_{M-N+1}=b_{M-N+1}^i b_{M-N+2}^i \cdots b_M^i$ 等字符串。计算每一个字符串相对于完整报文样本的出现频率。在统计时，如果一个字符串在一个报文中出现多次，只进行一次统计，因为此阶段处理的目的是获取候选关键词集合，而协议关键词是对协议格式或者报文字段的一种标识，往往不会在同一样本中重复出现。

在得到的字符串中，大部分字符串出现的频率较低，它们通常对应着报文中的变值字段或者是变值字段的子串，这些字段取值不确定甚至完全随机产生。协议格式提取希望确定的是相对稳定的协议报文构成结构，故设置阈值 T_{freq}，对这些出现频率较低的字符串进行过滤。将频率高于阈值的字符串作为候选关键词添加到候选关键词集合 G 中。

由于候选关键词集合 G 中的元素都是通过*N*-gram分词得到，其长度皆为 N，而实际协议报文中，协议关键词长度并不完全相同。对于长度大于 N 的协议关键词，可以由长度为 N 的子串合并得到，如HTTP协议中的关键词“Host”可以由“Hos”和“ost”合并得到。鉴于这种相关性，本文使用点间互信息对集合 G 中的候选词进行合并，以得到不同长度的协议关键词。

为了对候选关键词进行合并，首先，需要依据候选关键词集合 G 对报文样本进行标注，得到对应的候选关键词序列 $key\_seq_i=\{<w_1^i, f_1^i, pos_1^i>, <w_2^i, f_2^i, pos_2^i>, \cdots, <w_j^i, f_j^i, pos_j^i>\}$，其中 w_j^i 表示报文 m_i 中的第 j 个候选关键词，f_j^i 表示 w_j^i 在报文样本集合中出现的频率，pos_j^i 表示 w_j^i 在报文 m_i 中相对于报文起始位置的偏移。进而得到整个报文样本对应的候选词序列集合 $Seq=\{key\_seq_1, key\_seq_2, \cdots, key\_seq_i, \cdots\}$。

随后，基于互信息对集合 Seq 中的每个候选关键词序列 key_seq_i 中的协议关键词进行合并，具体步骤如下。

步骤1　从左至右遍历候选关键词序列 key_seq_i，寻找位置相邻且存在相同的连续 $N-1$ 个字符(即满足以下条件)的候选关键词：

(1)候选词 $w_j^i=b_g \cdots b_{g+l_j-1} \in G$，并且 $w_{j+1}^i=b_h \cdots b_{h+l_{j+1}-1} \in G$。

(2)候选词 w_j^i，w_{j+1}^i 位置相邻：$pos_{j+1}^i-pos_j^i=1$。

(3)候选词 w_j^i 的后 $N-1$ 个字符与 w_{j+1}^i 的前 $N-1$ 个字符相同：$b_{g+l_j-N+2} \cdots b_{g+l_j-1}=b_h \cdots b_{h+N-2}$。

步骤2　对于满足步骤1中条件的候选词 w_j^i，w_{j+1}^i，搜索点间互信息集合 PMI(初始为空)，若其中存在 $PMI(w_j^i, w_{j+1}^i)$ 则直接获取其值，若不存在则计算它们之间的互信息 $PMI(w_j^i, w_{j+1}^i)$ 并存入集合 PMI 中。得到候选词之间的互信息后，若 $PMI(w_j^i, w_{j+1}^i)$ 超过所设定的阈值 T_{PMI}，则合并候选词 w_j^i，w_{j+1}^i 的重叠部分得到新的候选词 w：

$$w=b_g \cdots b_{g+l_j-N+1} b_{g+l_j-N+2} \cdots b_{g+l_j-1} b_{h+N-1} \cdots b_{h+l_{j+1}-1}$$

步骤3　从 key_seq_i 中删除 w_{j+1}^i，将 w_j^i 值替换为 w。随后从 w_j^i 继续遍历更新后的 key_seq_i，重复步骤1~3直至遍历完 key_seq_i。

步骤4　由于协议关键词一般不会在同一报文中重复出现，故遍历 key_seq_i，删除在同一个关键词序列中出现多次的元素，得到最终的协议关键词序列以及更新后的关键词序列集合 Seq。

2.2.3　协议格式推断阶段

通过上述步骤能够得到每个报文样本对应的协议关键词，而相同格式的报文对应的关键词序列往往相同，因此可以依据协议关键词序列对报文样本进行聚类，将同种格式报文聚成一类，依据同类报文样本在结构上的相似性提取协议格式信息。本文提出使用扩展前缀树描述协议关键词序列，扩展前缀树中的每条路径代表一种关键词序列，节点对应协议关键词，节点之间的边代表报文样本中对应关键词之间的报文片段。采用扩展前缀树的描述方法，一方面便于实现对报文样本的初始聚类，另一方面便于实现分段的多序列比对。

2.2.3.1　构建扩展前缀树

前缀树(Prefix Tree)是一种有序多叉树结构，通常以字符串为输入，利用字符串的公共前缀来实现快速检索以及字符串匹配等操作。本文对前缀树进行扩展，将

协议关键词序列作为输入,协议关键词作为节点,按照顺序将协议关键词插入前缀树中,以得到的扩展前缀树描述报文样本对应的协议关键词序列。

为了构建扩展前缀树,在得到协议关键词序列集合 Seq 后,遍历集合 Seq ,将其中的元素依次插入树结构中。同时,为了排除样本中的噪声,在构建扩展前缀树的过程中记录每种关键词序列对应的报文样本总数,删除样本数小于阈值 T_{num} 的关键词序列。举例说明,对于包含表1所示元素的集合 Seq ,首先构建根节点“Start”表示起始位置,随后遍历集合 Seq ,依据其中的协议关键词序列将得到图3所示的扩展前缀树。

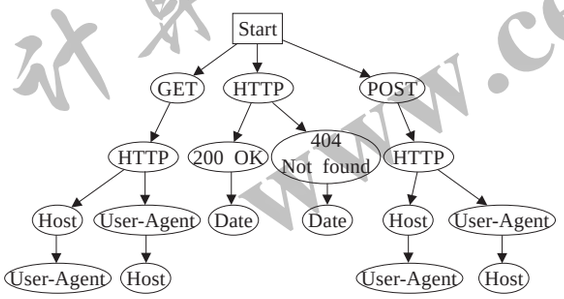

图3　扩展前缀树示例

扩展前缀树中的每条路径表示一种协议格式,图中共有6种协议格式。树中的每一个节点表示一个协议关键词,节点之间的边表示真实报文中对应协议关键词之间的报文片段。例如,扩展前缀树第1条和第2条路径中“GET”节点与“HTTP”节点之间的边,表示格式符合表1第1种或第3种关键词序列的对应的报文样本示例中协议关键词“GET”和“HTTP”之间的报文片段“admin.php”和“index.php”。

构建扩展前缀树的过程实际是对报文样本进行聚类以及对聚类子类中的报文进行分段的过程,同种格式的报文会汇聚在同一条路径上,并且依据节点对应的协议关键词分为多个片段,这种表示方式有利于后续提取每个报文片段的结构以及语义信息。

2.2.3.2　序列比对

对于得到的扩展前缀树,采用先深搜索遍历每一条路径,对同一路径上相邻两节点之间的报文(即扩展前缀树中的边)采用Needleman-Wunsch多序列比对算法进行比对,以提取详细的协议格式信息。包括协议字段取值类型(字符串、整数或二进制数等)、取值范围(包含常量字段、枚举字段、随机字段)等,并在扩展前缀树上做相应标注。

举例来看,若图3扩展前缀树中第1条路径所对应的报文如图4所示,其协议关键词序列为<“GET”,“HTTP”,“Host”,“User-Agent”>(图中“_”表示报文中本身包含的空格,“--”表示序列比对填充的空格)。通过多序列比对能够得到详细的格式信息,如第2个片段取值为一个浮点数后接回车换行符,且浮点数取值为“1.1”或“1.0”。

```
    | ----- |-|----|      |---|-|----|      |------|------|--------|           | --------------|
GET | _admin|.|php | HTTP |/1.|1|\r\n| Host |:_www.|foobar|.com\r\n| User-Agent | :_--Opera/9.20|
GET | _index|.|html| HTTP |/1.|1|\r\n| Host |:_www.|baidu |.com\r\n| User-Agent | :_--Opera/9.21|
GET | _test|.|jsp | HTTP |/1.|0|\r\n| Host |:_www.|google|.com\r\n| User-Agent | :_Mozilla/5.0-|
    | ------|-|----|      |---|-|----|      |------|------|--------|           | --------------|
```

图4　报文示例

通过这种分段的多序列比对,能够抽象出每个序列片段的报文结构,相对于PI项目等直接对整个报文进行序列比对,降低了时间复杂度。同时这种分段的格式提取方法,将报文依据协议关键词进行划段,也解决了多序列比对方法应用于结构复杂、报文过长的协议样本效果不佳的问题。

2.2.4　协议格式合并阶段

由于实际使用的通信协议往往具有较强的灵活性,完全依据协议关键词序列进行协议格式提取容易产生较多的冗余格式,即一种协议格式在推断时被划分为多种不同的协议格式。过多的格式冗余将导致推断结果的适用性降低。

一方面,一些报文字段的位置顺序是可变的。如HTTP协议中,“GET admin.php HTTP/1.1\r\n Host: www.foobar.com\r\n User- Agent: Opera/9.20”和“GET index.php HTTP/1.1\r\n User- Agent: Mozillia/5.0\r\n Host: www.baidu.com\r\n”两条报文实际上对应于一种协议格式,其中“Host”关键词和“User-Agent”关键词的先后顺序是可以变化的。但是在扩展前缀树中,两条报文对应的协议关键词序列不同,将被作为不同的协议格式。另一方面,一些报文中的协议关键词属于枚举类型。例如,HTTP请求方法字段可以在“GET”、“POST”、“HEAD”等协议关键词组成的集合中枚举。例如“GET admin.php HTTP/1.1\r\nHost: www.foobar.com\r\nUser-Agent: Opera/9.20”和“POST /eapi/pl/count HTTP/1.1\r\nHost: music.163.com\r\nUser- Agent: Mozilla/5.0”两条

表1　关键词序列集合的示例

编号	关键字序列	报文样本示例
1	<“GET”,“HTTP”,“Host”,“User-Agent”>	Get admin.php HTTP/1.1\r\n Host:www.foobar.com\r\nUser-Agent:Opera/9.20
2	<“POST”,“HTTP”,“Host”,“User-Agent”>	POST /eapi/pl/count HTTP/1.1\r\nHost:music.163.com\r\nUser-Agent:Mozilla/5.0
3	<“GET”,“HTTP”,“User-Agent”,“Host”>	Get index.html HTTP/1.1\r\nUser-Agent:Mozillia/5.0\r\nHost:www.baidu.com\r\n
4	<“POST”,“HTTP” ,“User-Agent”,“Host”>	POST /service/main/ucHTTP/1.1\r\nUser-Agent:Mozilla/5.0\r\nHost:my.csdn.net\r\n
5	<“HTTP”,“200 OK”,“Date”>	HTTP/1.1 200 OK\r\nDate:Sun,21 May 2017 01:51:25 GMT\r\n
6	<“HTTP”,“404 Not found”,“Date”>	HTTP/1.1 404 Not found\r\nDate:Sun,21 May 2017 01:51:25 GMT\r\n

报文仅仅请求方法不同,实际隶属于同一种协议格式。

因此需要对得到的扩展前缀树进行合并,本文提出以下合并策略:

(1)若扩展前缀树中两条路径所包含的节点种类完全相同,或某一条路径包含的协议关键词是另一条路径对应关键词的子集,并且两条路径上相同节点对应的边结构相似,则认为这两条路径对应着位置可变的协议格式,故将两者合并为一种格式(保留节点数较多的路径)。例如,图3扩展前缀树中路径1和路径2、路径5和路径6所包含的节点种类完全相同,只是部分节点位置不同,若两条路径中相同节点间的边对应的结构相似,则可删除其中任一条路径,得到如图5所示扩展前缀树。

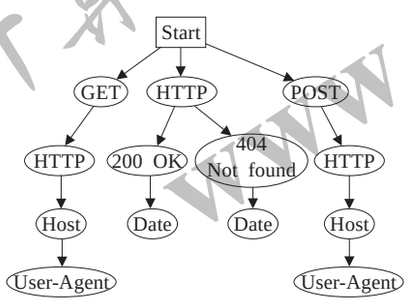

图5　第一种格式合并策略的示例

(2)若两条路径仅有一个节点不同,且与此节点相关的边在两条路径上结构相似,则认为这两条路径中的不同节点对应着同种协议格式的枚举字段的两种取值,故将两节点合并为一个节点,两条路径合并为一种协议格式。譬如,采用方法(1)对图3扩展前缀树进行合并后,最左边的路径和最右边的路径中仅有一个节点"GET"和"POST"不同,中间两条路径中仅有一个节点"200 OK"和"404 Not found"不同,则可实施合并,得到如图6所示的扩展前缀树。

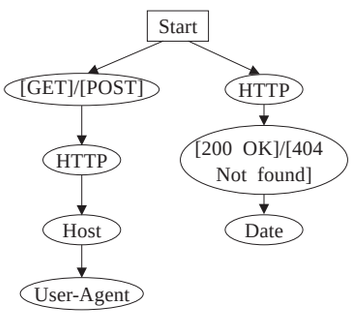

图6　第二种格式合并策略的示例

通过格式合并得到的最终扩展前缀树即代表了目标协议的详细格式,每一条路径对应一种协议格式。这种协议格式表示方式能够有效减少冗余,提高协议格式结果的实用性。

3　实验结果及分析

3.1　实验方法

为了对本文提出的方法进行验证,在重用Netzob[13]多序列比对代码的基础上,实现了原型系统EPT-PFI。实验中,选取文本协议HTTP和FTP与二进制协议DNS和NetBIOS作为实验对象。同时选取了现有的协议逆向工具Netzob、AutoReEngine进行对比试验。本文采用文献[7]中的准确率(Correctness)和精简率(Conciseness)作为评价指标对实验结果进行评估。准确率表示的是推断所得格式与实际格式相符的报文样本所占的比例,准确率越接近1说明推断结果越接近实际格式。精简率表示的是推断所得格式的种类数目与实际样本中包含的格式种类数的比值。由于是比值,精简率没有单位。精简率越接近1说明所得结果越接近真实格式。由于协议格式推断中常可能将同种格式报文划分为多种格式造成冗余,因此精确率越小说明得到的协议格式种类越少,冗余越少。

3.2　结果分析

本文在配置为2.93 GHz的CPU、4 GB内存、操作系统为Windows 7的PC上基于Python语言实现了论文中提出的方法。考虑到AutoReEngine关键字识别方法实用性强,准确率高,并且具有代表性[5],基于SPMF平台[14]提供的Apriori算法实现了AutoReEngine方法,与本文方法进行比较。

原型系统EPT-PFI设定阈值 T_{freq} 为0.6,设定阈值 T_{PMI} 为0.8,依据文献[15] N-gram分词中 N 取3。表2、表3是分别针对HTTP协议所获得的协议关键词(其中"_"表示空格)。分析实验结果:(1)从表2可以看出,EPT-PFI得到的协议关键词与实际基本吻合(所得关键词附带"\r\n"或"_",因为它们总是同时出现,所以被合并),但受限于样本的多样性,可能将某些可变字段被识别成了固定字段,如协议版本字段"HTTP/1.1",这是由于所捕获的样本中基本都是使用的1.1版本的HTTP协议。而其中的协议关键词"HTTP/1.1"两边没有携带空格换行等符号,是由于HTTP协议请求报文和应答报文中都包含版本号字段。(2)综合分析表2和表3可知,相对于AutoReEngine,EPT-PFI提取的协议关键词更加丰富,这是由于HTTP协议首部行中各个字段的位置是可变的,导致对应的关键词位置偏移较大。依据这些关键词结合扩展前缀树描述,能够获取首部行中各个字段的报文结构,得到更加详细的协议格式信息。

表2　EPT-PFI针对HTTP协议提取的协议关键词

编号	关键词	编号	关键词
1	GET_	7	\r\nAccept:
2	HTTP/1.1	8	\r\nCookie:
3	\r\nHost:	9	\r\nAccept-Encoding:gzip
4	\r\nUser-Agent:	10	\r\nAccept-Language:
5	\r\nConnection:Keep-alive:\r\n	11	POST_
6	\r\nReferer:	12	_200_OK\r\n

表3　AutoReEngine针对HTTP协议提取的协议关键词

编号	关键词	编号	关键词
1	GET_	4	\r\nHost:
2	HTTP/1.1	5	_200_OK\r\n
3	POST_	…	…

如图7、图8分别为采用EPT-PFI、Netzob、AutoReEngine对4种目标协议进行实验所得结果。从实验结果可以看出:(1)EPT-PFI在准确率上高于Netzob,稍低于AutoReEngine。AutoReEngine准确率几乎都达到了100%,这是因为AutoReEngine只保留了位置相对固定的协议关键词,对于字段位置可变的协议关键词都会被Apriori算法剪枝,如HTTP协议中的“Host”、“Connection”等字段由于在报文样本中出现的位置变化较大而被忽略。因此,AutoReEngine只能够提取出位置相对固定的字段关键词,但无法获取详细的格式信息。(2)在精简率上Netzob明显高于EPT-PFI和AutoReEngine,说明所得协议格式冗余较多,这是由于Netzob是在多序列比对的基础上进行报文聚类的,可能将同种格式报文划分为多种格式。而EPT-PFI由于采用了基于扩展前缀树的格式合并,其精简率略优于AutoReEngine。综合来看,Netzob对于文本协议效果明显优于二进制协议,而本文方法对于文本协议和二进制协议都能取得较好结果。

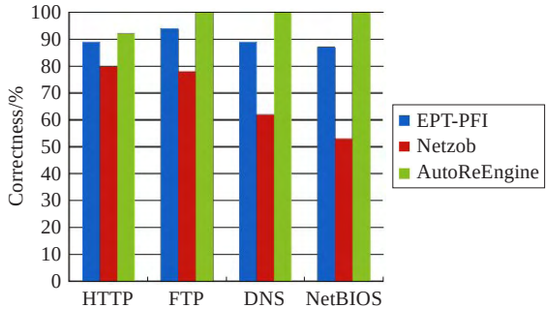

图7　正确率实验结果

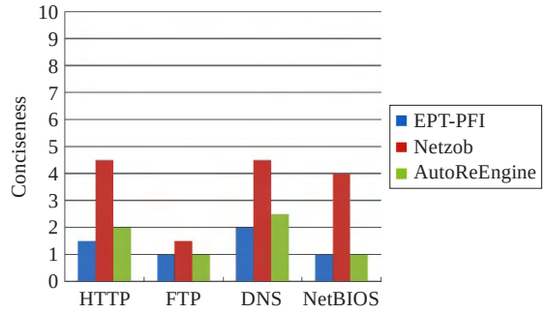

图8　精简率实验结果

通过上述实验分析,可以看出原型系统MI-PFE能够较好地完成文本协议以及二进制协议的格式推断工作。

4　总结

本文提出了一种基于扩展前缀树的协议格式推断方法,通过*N*-gram分词结合互信息获取协议关键词,依据协议关键词序列构建扩展前缀树,在扩展前缀树描述的基础上进行分段多序列比对,最终通过对报文结构的扩展前缀树进行合并得到最终协议格式。实验表明,本方法能够适用于文本协议和二进制协议,并且与现有的协议格式推断工具相比具有一定优势。

参考文献:

[1] 罗建桢,余顺争,蔡君.基于最大似然概率的协议关键词长度确定方法[J].通信学报,2016,37(6):119-128.

[2] Sija B D,Goo Y H,Shim K S,et al.A survey of automatic protocol reverse engineering approaches,methods,and tools on the inputs and outputs view[J].Security & Communication Networks,2018:1-17.

[3] Gascon H,Wressnegger C,Yamaguchi F,et al.Pulsar:Stateful black-box fuzzing of proprietary network protocols[M]//Security and Privacy in Communication Networks.Germany:Spring International Publishing,2015:330-347.

[4] Duchêne J,Guernic C L,Alata E,et al.State of the art of network protocol reverse engineering tools[J].Journal of Computer Virology & Hacking Techniques,2017:1-16.

[5] Narayan J,Shukla S K,Clancy T C.A survey of automatic protocol reverse engineering tools[J].ACM Computing Surveys,2015,48(3):1-26.

[6] Marshall B.Protocol information project[EB/OL].(2004-10-05)[2017-12-24].http://www.4tphi.net/~awalters/PI/PI.html.

[7] Cui W,Kannan J,Wang H J.Discoverer:Automatic protocol reverse engineering from network traces[C]//16th USENIX Security Symposium.Boston:USENIX Association,2007:199-212.

[8] Krueger T,Kraemer N.PRISMA:Protocol inspection and state machine analysis[J].Journal of the American Chemical Society,2015,98(25):8101-8107.

[9] Zhang Zhuo,Zhang Zhibin,Lee P P C,et al.ProWord:An unsupervised approach to protocol feature word extraction[C]//Proceedings IEEE INFOCOM,2014:1393-1401.

[10] Luo J Z,Yu S Z.Position-based automatic reverse engineering of network protocols[J].Journal of Network & Computer Applications,2013,36(3):1070-1077.

[11] 黎敏,余顺争.抗噪的未知应用层协议协议格式最佳分段方法[J].软件学报,2013,24(3):604-617.

[12] 李伟明,张爱芳,刘建财,等.网络协议的自动化模糊测试漏洞挖掘方法[J].计算机学报,2011,34(2):242-255.

[13] Bossert G,Hiet G.Towards automated protocol reverse engineering using semantic information[C]//ACM Symposium on Information,Computer and Communications Security,2014:51-62.

[14] Fournier-Viger P,Gomariz A,Gueniche T,et al.SPMF:A Java open-source pattern mining library[J].Journal of Machine Learning Research,2014,15(1):3389-3393.

[15] Wang Y,Zhang Z,Yao D D,et al.Inferring protocol state machine from network traces:A probabilistic approach[C]//International Conference on Applied Cryptography and Network Security.Berlin:Springer-Verlag,2011:1-18.