一种基于域知识的协议状态机主动推断算法

王 辰 吴礼发 洪 征 郑成辉 庄洪林
(解放军理工大学指挥信息系统学院 南京 210007)

摘 要 现有基于 L^* 算法的协议状态机主动推断方法忽略了协议特有的域知识,将协议报文抽象为相互独立、无意义的符号,并完全随机地生成测试样本进行状态机等价判定,导致产生大量的无效询问和测试样本,在真实网络环境下推断效率较低。在 L_M^+ 算法的基础上提出了一种基于域知识的协议状态机主动推断算法 L_N^+,其改进主要体现在:依据会话样本集提取各报文之间的强顺序约束关系来过滤无效的输出询问,构建会话样本集对应的扩展前缀树接受器(Extended Prefix Tree Accepter,EPTA)对输出询问进行预响应,提出了一种基于正例样本变异的等价询问近似判定算法以提升寻找反例的效率。实验结果表明,L_N^+ 算法能够大幅提高推断效率,并且具有与 L_M^+ 算法相同的推断准确度。

关键词 L^* 算法,协议状态机,主动推断,域知识,推断效率
中图法分类号 TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.12.050

Domain-specific Algorithm of Protocol State Machine Active Inference

WANG Chen WU Li-fa HONG Zheng ZHENG Cheng-hui ZHUANG Hong-lin
(College of Command Information System,PLA Univeristy of Science and Technology,Nanjing 210007,China)

Abstract Existing protocol state machine inference approaches based on algorithm L^* are inefficient owing to ignorance of protocol-specific knowledge. As the protocol messages are abstracted as the independent and insignificant symbols,and test samples are completely generated randomly in equivalence query,invalid queries and test samples are inevitable. A protocol state machine active inference algorithm named L_N^+ was proposed,which improves the algorithm L_M^+ in three aspects. Firstly,L_N^+ filters the invalid output query according to the constraint on strict order,which is extracted from conservation samples. Secondly,L_N^+ constructs the extended prefix tree accepter(EPTA) corresponding to the sample set and answers the output query in advance. Thirdly,a new proposed strategy to find counterexamples more effectively is applied to judge the equivalence query based on positive sample mutation. Experimental results show that L_N^+ improves the inference efficiency greatly and achieves the same accuracy as algorithm L_M^+.

Keywords Algorithm L^*,Protocol state machine,Active inference,Domain-specific knowledge,Inference efficiency

1 引言

协议状态机定义了协议实体间的通信报文序列以及交互行为,在安全领域有着广泛的应用[1-3]。例如,针对未知状态型协议的模糊测试需要依据协议的状态转移信息生成测试报文序列,才能避免因状态不匹配而产生大量的无效测试用例。

根据推断过程中是否需要与协议实体交互来指导推断,状态机推断可分为被动推断和主动推断两类。被动推断采用离线分析会话样本集的方式,基于状态融合的思想被动地推断状态机[4-6],具有实现简单、推断效率高的优点,但存在过融合的问题,并且推断结果的完备性取决于样本集的覆盖程度[7]。为了克服被动推断方法的缺陷,研究者尝试基于请求询问(query)和应答的机制与协议实体在线交互,主动推断协议状态机。

代表性的状态机主动推断算法主要有两个:一个是 Dupont 等人提出的 QSM(query-driven state merging)[9] 算法,它虽然解决了过融合问题,但是推断结果仍然依赖于正例样本集的覆盖程度,且难以保证推断的状态机是最精简的;另一个是 Angluin 提出的学习型算法 L^*[10],它能够保证在多项式时间内推断得到的确定型 Moore 机是最小且完备的。通过成员资格询问(membership query)和等价询问(equivalence query)与假设可以对其作出回答的 Oracle 交互,依据 membership query 的响应构建闭合连续的观察表(Observation Table,OT),进而生成对应的候选状态机 M_M,依据等价询问(equivalence query)的响应判断 M_M 是否与真实的状态机一致,若是则终止推断;否则返回可以表现二者差异的反例,重新推断。Shahbaz[11] 以确定型 Mealy 机为推断目标,改进了反例(真实状态机与推断)的处理策略,提出了相应的 L_M^+ 算法,减少了两种 query 的数量。Cho 等人[12] 基于协议格式、语义逆向提取方法 Dispatcher[13] 首次将 L_M^+ 算法应用到协议状

到稿日期:2014-11-26 返修日期:2015-02-05 本文受国家自然科学基金项目(611032253),江苏省自然科学基金项目(BK2011115)资助。
王 辰(1990-),男,硕士生,主要研究方向为网络安全;吴礼发(1968-),男,教授,博士生导师,主要研究方向为网络安全;洪 征(1979-),男,副教授,硕士生导师,主要研究方向为网络安全;郑成辉(1977-),男,讲师,主要研究方向为网络安全;庄洪林(1963-),男,高级工程师,主要研究方向为网络安全。

态机推断领域，提出了 query 缓存、并行化和预测机制，以提升推断效率，并且借鉴了 L^* 算法中的 equivalence query 近似判定方法，基于输入报文类型集随机生成测试样本来判定推断的候选状态机是否与真实状态机近似等价。然而，以上基于 L^* 算法的研究均没有捕获和学习目标协议系统间交互的会话样本集，忽略了协议状态机推断领域特有的域知识，主要体现在以下 3 点：

(1)将各类型的报文抽象为相互独立、无意义的符号，忽略了各报文之间的顺序约束关系，产生了大量的不被接受的无效 output query；

(2)忽略了可以捕获会话样本集来对部分 output query 直接做出响应，没有必要将所有 output query 均发送至协议实体，询问输出结果；

(3)完全随机地生成测试样本，忽略了反例样本与正例样本在结构上的相关性和测试样本的有效域是在改变的，导致生成大量无效的、重复的测试样本，降低了等价判定效率。

将大量无意义的 output query 和测试样本实例化为协议实体可接受的报文序列，并通过网络发送给协议实体询问响应，需要较大的时间耗费，而且推断效率过低。本文首先捕获了大量的会话样本集并进行充分学习，以获取目标协议特有的域知识，进而在 L_M^+ 算法的基础上提出了一种基于域知识的协议状态机主动推断算法 L_N^+，并实现了相应的原型系统，最后通过实验验证了该算法的有效性。

2 问题描述与定义

2.1 问题描述

因为协议系统是一个依据输入进行响应的交互系统，所以采用带输出的 Mealy 机作为协议状态机的形式化描述模型，如定义 1 所示。

定义 1(Mealy 机) 一个 Mealy 机[12]可定义为一个六元组 $M=(Q,I,O,\delta,\lambda,q_0)$，其中：$Q$ 是有限非空状态集，$q_0\in Q$ 是初始状态，I 是有限输入符号集，O 是有限输出符号集，δ：$Q\times I\to Q$ 是状态转移函数，λ：$Q\times I\to O$ 是输出函数。

需要说明的是，本文的研究只针对确定型有限状态机，即定义 1 中的状态转移函数 δ 为一对一或多对一的映射。进一步可将定义 1 中的状态转移函数 δ 和输出函数 λ 在状态 q 下的可接受输入从单个字符 $i\in I$ 扩展至字符序列 $\omega\in I^*$，例如对于状态 $q_1\in Q$，输入字符序列 $\omega=i_1\cdots i_k\in I^*$，状态机逐步转移至 $q_{j+1}=\delta(q_j,i_j)$，$1\leqslant j\leqslant k$，最后到达状态 $q_{k+1}=\delta(q_1,\omega)$，输出字符序列 $\lambda(q_1,\omega)=o_1\cdots o_k$，其中 $o_j=\lambda(q_j,i_j)$，称 $i_1/o_1\cdots i_k/o_k$ 是在状态 q_1 下的一个 I/O 序列。

对于上文提到的状态机完备性，需要说明的是，不存在任何自动化的方法可以保证 I 的完备性，只要对于给定的 I，推断所得状态机中的 δ 对 $\forall q\in Q,i\in I$ 是定义完全的，则认为该 Mealy 机是完备的。

由于有限状态机推断和正则语言学习是同构的，协议状态机推断可以看作文法推断(Grammatical Inference)的一种特殊应用场景[15]。所以需要在协议报文格式、语义提取的基础上，对协议状态机推断中捕获的报文样本进行报文聚类和符号抽象，提取出对应的有限输入/输出符号集 I/O。本文在已经完成协议报文格式和语义的逆向提取工作[15,16]的前提下，重点对协议状态机推断进行研究。需要说明的是，本文基于的报文样本都是未加密的，但是所提方法仍适用于解密后的加密协议。

2.2 L_N^+ 算法基础定义

为便于后文阐述，本节首先给出 L_N^+ 算法中所涉及的关键概念的定义。

L_N^+ 算法维护了一个观察表(Observation Table，OT)来记录输出询问的结果，如定义 2 所示。

定义 2(观察表) 一个 OT[11]可定义为一个三元组 (S,E,T)，其中 S 为基于 I^* 的前缀闭合集，E 为基于 I^+ 的后缀闭合集，T 为 $(S\cup S\cdot I)\times E$ 至 O^+ 的映射，其中 $S\cdot I=\{s\cdot a\mid s\in S,a\in I\}$。另外，对于 $\forall s\in S\cup S\cdot I,e\in E,x\in O^+$，$(T(s,e)=x)\Rightarrow|e|=|x|$。

关于 OT 有以下两个性质。

性质 1(等价性[10]) 对 $\forall s,t\in S\cup S\cdot I$，当且仅当对所有 $e\in E$，均满足 $T(s,e)=T(t,e)$，那么称 s 等价于 t，用 $s\cong_E t$ 表示，$[s]$表示所有与 s 等价的行。

性质 2(闭合性[10]) 若对 $\forall t\in S\cdot I$，$\exists s\in S$，满足 $s\cong_E t$，则称 OT 闭合。

L_N^+ 算法依据协议实体对输入的响应，来实现 L_M^+ 算法中假设可以对以下两种请求询问(query)作出回答的 Oracle 的功能。为与 L_M^+ 算法保持一致，下文仍继续使用“Oracle”。

(1)输出询问(output query)[10]：由符号序列 $s\cdot e$ 组成，其中 $s\in S\cup S\cdot I,e\in E$，向 Oracle 询问状态机在状态[s]下接受符号序列 e 后的输出，即 $T(s,e)=\lambda([s],e)$。

(2)等价询问(equivalence query)[10]：用于向 Oracle 询问推断的候选 Mealy 机 M_M 是否与真实状态机一致。

一个闭合的 $OT(S,E,T)$ 对应一个候选 Mealy 机 M_M，如定义 3 所示。

定义 3(候选 Mealy 机) 一个候选 Mealy 机 M_M[12]可定义为六元组 $(Q_M,I,O,\delta_M,\lambda_M,q_{0M})$，其中：有限非空状态集 $Q_M=\{[s]\mid s\in S_M\}$；初始状态 $q_{0M}=[\varepsilon]$；状态转移函数 $\delta_M([s],i)=[s\cdot i]$，$\forall s\in S_M,i\in I$，表示在状态[s]下输入 i 后的目标状态；输出函数 $\lambda_M=([s],i)=T_M(s,i)$，$\forall i\in I$。

3 基于 L_N^+ 算法的协议状态机主动推断

3.1 L_N^+ 算法概述

L_N^+ 算法的执行流程如图 1 所示。

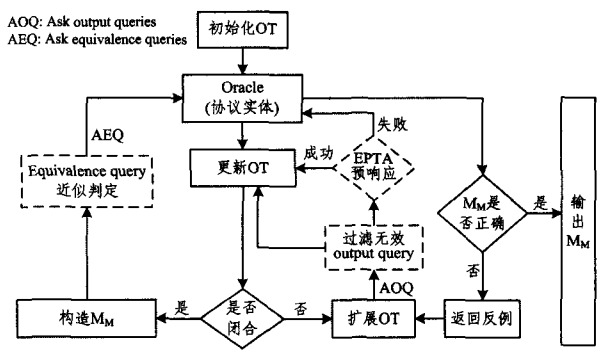


图 1 L_N^+ 算法流程

其主要有以下几个步骤：

(1)初始化 OT，令 $S=\varnothing$，$S\cdot I=\{\varepsilon\}$，$E=I$，生成 output query 并发送至 Oracle，根据返回结果更新 OT。

(2)检查 OT 的闭合性：当 OT 闭合时，进入步骤(6)；当 OT 不闭合时，寻找 $t\in S\cdot I$，对于所有 $s\in S$，均不满足 $t\cong_E s$，将 t 移至 S，扩展 OT，生成 output query，向 Oracle 询问输出结果。

(3)通过学习会话样本集提取各报文类型之间的强顺序约束关系，制定相应的过滤规则，若判定某 output query 无效，则直接响应，否则不做处理。

(4)对步骤(3)中判定为有效的 output query 利用依据会话样本集构建的 EPTA 进行预响应，若响应成功则直接更新 OT，否则再发送至 Oracle，询问输出结果。

(5)重复步骤(2)—步骤(4)，直至 OT 闭合。

(6)依据闭合的 OT 构造对应的候选 Mealy 机 M_M，图 2 展示了依据表 1 构造的 M_M。由定义 3 可知：$Q_M=\{[\varepsilon],[a]\}$；初始状态 $q_{0M}=[\varepsilon]$；状态转移函数 $\delta_M([\varepsilon],a)=[a]$，$\delta_M([\varepsilon],b)=[b]\cong_E[a]$，$\delta_M([a],a)=[a\cdot a]\cong_E[a]$，$\delta_M([a],b)=[a\cdot b]\cong_E[\varepsilon]$；输出函数 $\lambda_M=([\varepsilon],a)=T_M(\varepsilon,a)=x$，$\lambda_M=([\varepsilon],b)=T_M(\varepsilon,b)=x$，$\lambda_M=([a],a)=T_M(a,a)=y$，$\lambda_M=([a],b)=T_M(a,b)=x$。

表 1　闭合 OT 示例

T		E	
		a	b
S	ε	x	x
	a	y	x
S·I	b	x	x
	a·a	y	x
	a·b	x	x

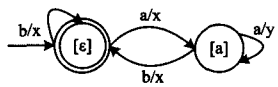


图 2　由表 1 构造的 M_M

(7)依据提出的基于正例样本变异的 equivalence query 近似判定算法(Approximate Judgement Based on Positive Sample Mutation，APSM)，判定候选状态机 M_M 是否与真实状态机等价，若等价则输出该 M_M，否则返回一个反例 c，依据步骤(8)对其进行处理。

(8)令反例 $c=u\cdot v$，其中 $u\in S\cup S\cdot I$ 是 c 的最长前缀，且满足 $\lambda_M(q_{0M},u)$ 与真实状态机输出相同，如此保证了 v 为长度最小的区分序列；然后扩展 OT，将 v 和其所有后缀加入到 E 中，既保证了 OT 的闭合性，又减小了对 E 的扩展，从步骤(2)开始，重新推断，直到输出正确的 M_M。

$L_N^{\dagger}$ 算法对 $L_M^{\dagger}$ 算法的改进主要集中在上述算法流程中的步骤(3)、步骤(4)和步骤(7)，如图 1 中虚线方框所示，分别对应于基于顺序约束关系的 output query 过滤机制、基于 EPTA 的 output query 预响应机制和基于正例样本变异的 equivalence query 近似判定算法，3.2 节—3.4 节将分别对其进行详细阐述。

3.2　基于强顺序约束关系的 output query 过滤机制

基于强顺序约束关系的 output query 过滤机制是通过充分学习会话样本集 S 来实现的。在学习前需要对 S 进行符号抽象，经处理之后，一个会话样本 $s\in S$ 可描述为 $(\cdots i_j/o_j\cdots i_k/o_k\cdots i_m/o_m\cdots)$，其中 $i\in I$，$o\in O$，$1\leqslant j,k,m\leqslant|I|$，对应的输入报文符号序列 $s_I=(\cdots i_j\cdots i_k\cdots i_m\cdots)$。首先给出单个会话样本上的弱顺序约束关系，如定义 4 所示。

定义 4(单个会话样本上的弱顺序约束关系)　对于一个会话样本的输入报文符号序列 $s_I=(\cdots i_j\cdots i_k\cdots i_m\cdots)$，表示 i_j 在 s_I 中的顺序位于 i_k 和 i_m 之前，则称样本 s 上 i_j，i_k，i_m 具有弱顺序约束关系，可表示为 $i_j\rightarrow i_k$，$i_j\rightarrow i_m$，$i_k\rightarrow i_m$ 或 $i_j\rightarrow i_k\rightarrow i_m$。

若输入报文类型数为 n，则整个输入报文符号序列集 S_I 上的弱顺序约束关系集可以用一个 $n\times n$ 的矩阵表示，称为输入符号集 I 上的弱约束关系邻接矩阵，如定义 5 所示。

定义 5(有限输入符号集 I 上的弱顺序约束关系邻接矩阵)　矩阵第 j 行第 k 列 v_{jk} 表示满足 $i_j\rightarrow i_k$ 的会话样本 $s_I\in S$ 的数目，可形式化表示为 $v_{jk}=\{|s_I||i_j,i_k\subset s_I\in S\&i_j\rightarrow i_k\}$。

基于定义 5 所示的邻接矩阵，可以遍历 S 中所有的会话样本统计得到存在弱顺序约束关系的报文类型对数目。进一步地，需要获取具有强顺序约束关系的报文类型对，如定义 6 所示。

定义 6(有限输入符号集 I 上的强顺序约束关系 R)　如果弱顺序约束关系邻接矩阵中 $v_{jk}\neq0$，且 $v_{kj}=0$，则认为报文类型对 (i_j,i_k) 具有强顺序约束关系，可形式化表示为：$R=\{(i_j,i_k)|\ i_j,i_k\in I\ \&v_{jk}\neq0,v_{kj}=0\}$。

定义 5 和定义 6 基于概率统计的思想，表示仅满足 $i_j\rightarrow i_k$ 且不满足 $i_k\rightarrow i_j$ 的报文类型对 (i_j,i_k) 才具有强顺序约束关系。进一步地，可将 R 扩充为报文类型集间的强约束关系，即 $R=\{(i_j,i_m),(i_k,i_m),(i_j,i_n),(i_k,i_n)\}\Leftrightarrow R=\{(\langle i_j,i_k\rangle,\langle i_m,i_n\rangle)\}$。但需要说明的是，为避免因样本集规模受限而导致统计结果不准确的问题，需要结合协议报文的语义信息，人工对提取的 R 作进一步的确认，以保证推断的准确度。

在获取并确认了有限输入符号集 I 上的强顺序约束关系 R 后，利用正则表达式自动化地生成无效 output query 的过滤规则。图 3 以 SMTP 协议为例，展示提取的 R_{SMTP} 和其对应的 6 条过滤规则，其中 w 表示 output query，各输入符号对应的输入报文类型如 4.2.1 节中表 2 所列。以依据 $(\langle1,2,3,4,5,6\rangle,7)$ 制定的过滤规则①为例，它表示如果 w 包含报文类型 $i\in\langle1,2,3,4,5,6\rangle$ 和报文类型 7，但不匹配正则表达式"[1−6]+[7]$"，则判定该 output query 无效。该正则表达式中，"[1−6]+"表示 1 到 6 中的一个数字出现一次或多次，"[7]$"表示以 7 终止。依据表 2 可知，规则①的含义为：所有 w 必须以 QUIT 报文结束。

提取到的SMTP协议的强顺序约束关系集：
$R_{SMTP}=\{(\langle1,2,3,4,5,6\rangle,7),(1,\langle2,3,4,5,6,7\rangle),(2,\langle3,4,5,6,7\rangle),(3,\langle4,5,6,7\rangle),(4,\langle5,6,7\rangle),(5,\langle6,7\rangle)\}$
依据R_{SMTP}制定的6条过滤规则：
① if one of ⟨1,2,3,4,5,6⟩ and 7 were in w
and w didn't match "[1-6]+[7]$" then it's invalid
② if 1 and one of ⟨2,3,4,5,6,7⟩ were in w
and w didn't match " ^[1][2-7]+ "then it's invalid
③ if 2 and one of ⟨3,4,5,6,7⟩ were in w
and w didn't match "^[1][2][3-7]+" then it's invalid
④ if 3 and one of ⟨4,5,6,7⟩ were in w
and w didn't match "^[1][2][3][4-7]+" then it's invalid
⑤ if 4 and one of ⟨5,6,7⟩ were in w
and w didn't match "^[1][2][3][4][5-7]+" then it's invalid
⑥ if 5 and one of ⟨6,7⟩ were in w
and w didn't match "[1][2][3][4][5][6-7]+" then it's invalid

图 3　SMTP 协议 output query 过滤规则

由于无效的 output query 经过滤后不会发送至 Oracle，因此需要在过滤的同时对其作出响应，以更新 OT。但是提取到的有限输出符号集 O 只包含对有效输入的响应，所以需

要对 O 作出扩展，增加输出符号 Ω，响应无效 output query "$s \cdot e$"，其中 $s \in S \cup S \cdot I, e \in E$，更新 $T(s,e)=\Omega$。

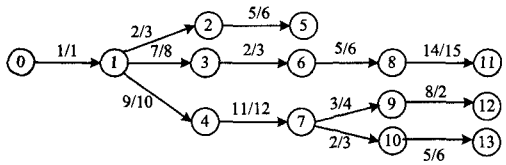
3.3 基于 EPTA 的 output query 预响应机制

由定义 1 可知，输出符号并不影响状态迁移，因此我们在传统的 PTA(Prefix Tree Accepter)中增加输出符号序列以得到扩展前缀树接受器 EPTA。EPTA 本质上是一个多叉树，可看作为一个初始状态机，每个分支代表了一个会话样本对应的抽象符号序列。

图 4(b)展示了由图 4(a)中符号抽象的会话样本集 S 构造对应 EPTA(S)的示例。初始状态为根节点，用阿拉伯数字 0 标识。状态机从初始状态开始接受输入，如果输入符号序列所到达的状态在原状态机中不存在，则创建一个新状态，以阿拉伯数字唯一标识，并添加状态转移过程中相应的输出信息；如果已存在，则转移至该状态，接受下一个输入，重复上述过程，直到 EPTA 接受样本集 S 中的所有输入符号序列。

$S=\{$ (1/1, 2/3, 5/6),
(1/1, 7/8, 2/3, 5/6, 14/15),
(1/1, 9/10, 11/12, 2/3, 5/6),
(1/1, 9/10, 11/12, 3/4, 8/2)
$\}$

(a)符号抽象的会话样本集



(b)构造的 EPTA(S)

图 4 EPTA 构造示意图

EPTA 构造完成之后，对其采用深度优先搜索的遍历方式来对发送至 Oracle 的 output query 进行预响应。假设 output query 为(1·9,11)，表示询问在状态[1·9]下输入"11"的输出响应。以图 4(b)为例，状态[1·9]对应于从初始状态 0 下输入符号序列"1·9"后转移至的状态 4，遍历图 4(b)可知在状态 4 下输入"11"的输出响应为"12"，则成功响应，更新 T(1·9,11)=12。若 output query 为(1·4,11)，不在该 EPTA 遍历范围，则响应失败，发送至 Oracle，询问输出。同时记录该 output query 和其输出，更新 EPTA，避免后续过程中对相同 output query 仍预响应失败的问题，进一步减少发送至 Oracle 的 output query 数目。

3.4 基于正例样本变异的 equivalence query 近似判定算法

对每个候选 Mealy 机 M_M 进行等价判定的 equivalence query 由若干个测试样本 s_{test} 组成，原始 $L_M^\sharp$ 算法基于 I 随机生成平均分布的符号序列测试样本 $s_{test} \in I^*$，发送至 Oracle，比较其输出是否与 M_M 对该测试样本的输出相同，如果不相同，则为反例。Angulin[10] 已经证明了如果在发送 r_i 个 s_{test} 后均没有产生反例，则认为该 M_M 是 M 的 ε 近似的概率大于或等于 $1-\delta$，输出 M_M。

$$r_n=\frac{1}{\varepsilon}\left(\log\frac{1}{\delta}+(\log 2)(n+1)\right) \tag{1}$$

式中，ε 为准确度(accuracy)，表示每次生成的随机符号序列是反例的概率不大于 ε；δ 为置信度(confidence)；n 为已经生成的候选 Mealy 机 M_M 的个数。虽然式(1)给出了当一直没有寻找到反例时算法的终止条件，但是 $L_M^\sharp$ 算法基于输入符号集 I 完全随机地构造测试样本来对每次生成的 M_M 进行判定的方法存在以下两点局限：

(1)忽略了反例样本和正例样本结构上的相关性：反例样本的意义在于寻找之前推断过程中被遗漏的状态，即在某个已存在的状态 $q \in Q_M$，接受输入 $i \in I$，转移至状态 $q' \notin Q_M$。所以反例样本和正例样本肯定具有一部分共同的前缀。

(2)忽略了测试样本的有效域是改变的：第 n 个 M_M 可接受的样本集包含前 $n-1$ 个 M_M 可接受的样本集，即对于前 $n-1$ 个 M_M 不是反例的测试样本，在第 n 次等价判定过程中必然也不是反例。

基于上述分析，本文提出了一种基于正例样本变异的 equivalence query 近似判定算法 APSM，以提升 equivalence query 的判定效率，如算法 1 所示。

算法 1 equivalence query 近似判定算法 APSM

输入：正例样本集 S_I，有限输入符号集 I，已经生成的测试样本集 T_{all}，已经生成的 M_M 数 n

输出：M_M or A counterexample

```
APSM (S_I, I, S_all, n)
1. for (N_t=0, N_t<r_n, N_t++)
2.     Random select s∈S_I and i∈I;
3.     Variate(s,i) ={Insert(s,i) |Replace(s,i) |Delete(s,i)}
4.     while Variate(s,i) S_all do s_test=Variate(s,i);
5.     S_all←s_test;
6.     Send s_test to Oracle;
7.     if s_test find a discepancy between M_M and M
8.         then s_test is a cunterexample, return s_test;
9.         break;
10.    else N_t++;
11.    end if
12. end for
13. return M_M
```

算法 1 的输入中正例样本集 S_I 是对 3.2 节中构建 EPTA 的样本集进行提取输入序列和去重预处理得到的。算法首先根据已经生成的 M_M 数 n 确定本次判定需要测试样本 s_{test} 的最大数目 r_n(第 1 行)。然后充分考虑反例样本和正例样本存在共同前缀的结构相关性，随机选择一个正例样本 $s \in S_I$ 和输入符号 $i \in I$，从插入、替换和删除操作中随机选择一种变异方式(第 2,3 行)，对其进行变异。以 s=(1,7,2,5,14)为例，s_{test}=Insert(s,i)=(1,7,i,2,5,14)，减小了生成的测试样本 s_{test} 均不被真实状态机和 M_M 接受的概率，从而增大了其寻找二者差异的可能性。同时充分利用对前 $n-1$ 个 M_M 不是反例的测试样本在第 n 次等价判定过程中必然也不是反例这一特性，保证生成的测试样本 s_{test} 在之前的判定过程中从未生成过，即 $s_{test} \notin T_{all}$，并将其存储至已经生成的测试样本集 T_{all} 中(第 4,5 行)，避免生成已在之前的判定过程中被证明不是反例的无意义的测试样本。最后将 s_{test} 发送至 Oracle，比对其输出是否与 M_M 一致，如果不一致，则 s_{test} 为反例，输出 s_{test}；否则重新生成测试样本，若在生成 r_n 个测试样本后，仍未产生反例，则判定该 M_M 是真实状态机 M 的 ε 近似的概率大于或等于 $1-\delta$，输出 M_M(第 6－13 行)。

4 实验与分析

4.1 原型系统

为了验证 $L_N^\sharp$ 算法的有效性，本文在 Ubuntu12.04 系统下，基于协议逆向分析平台 Netzob(0.4.1)[18] 中的语法推断

模块实现了原型系统。该系统由报文抽象模块、L_N^+算法模块、交互代理模块和服务器端目标程序组成，架构如图5所示，其中各个模块的功能如下。

报文抽象模块：负责将PCAP文件中的输入、输出报文序列分割为单个报文。依据报文格式、语义描述对报文进行聚类，提取出各类型报文对应的输入/输出符号集I/O。

L_N^+算法模块：实现第3节论述的L_N^+算法。

交互代理模块：负责将EPTA预响应失败的output query和equivalence query近似判定过程中生成的测试样本符号序列进行报文实例化，以及将服务器端目标程序对输入报文的响应报文符号化后反馈给L_N^+算法模块，更新OT。

服务器端目标程序：响应输入报文。

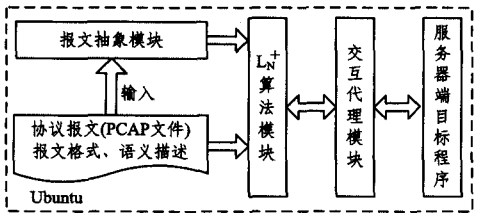


图5 原型系统架构

4.2 实验结果与分析

本文选用两种公开的文本协议(SMTP[17]、POP3[19])为目标协议，协议实体客户端程序为Windows 7操作系统下的Foxmail 7.2.5，服务器设置为smtp.163.com和pop.163.com，不采用SSL加密。在测试过程中，首先利用协议逆向分析平台Netzob的词法推断模块分析两种协议的报文格式和语义信息；然后在线捕获客户端与服务器端的交互报文，随机选取500个会话报文作为会话样本集；最后应用原型系统推断得到协议状态机。

4.2.1 输入、输出报文类型提取

学习会话样本集并抽象得到的SMTP协议的输入报文类型如表2所列。

表2 SMTP协议输入报文类型的抽象符号

输入符号	输入报文类型	输入符号	输入报文类型
1	HELO/EHLO	5	Content
2	MAIL FROM	6	RSET
3	RCPT TO	7	QUIT
4	DATA		

SMTP协议的输出报文为服务器端的响应报文，参考RFC821[17]，提取到的输出报文和其对应的含义如表3所列。

表3 SMTP协议输出报文类型

响应报文	含义
221	服务关闭传输信道
250	要求的邮件操作完成
354	开始邮件输入

表4展示了POP3协议输入报文类型对应的抽象符号。

表4 POP3协议输入报文类型的抽象符号

输入符号	输入报文类型	输入符号	输入报文类型
1	USER	5	UIDL
2	PASS	6	RETR
3	STAT	7	DELE
4	LIST	8	QUIT

提取到的POP3协议的输出报文类型仅有“+OK”，用“1”表示。

4.2.2 推断准确度

在4.2.1节提取输入、输出报文类型的基础上，应用原型系统，设置式(1)中的参数$\varepsilon=0.02$，$\delta=0.05$，推断得到SMTP与POP3协议的状态机分别如图6和图7所示。

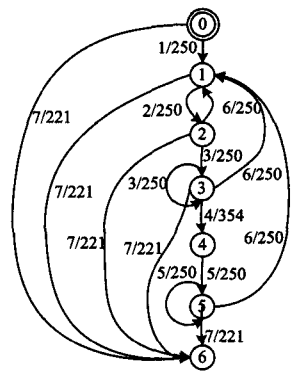


图6 推断得到的SMTP协议状态机

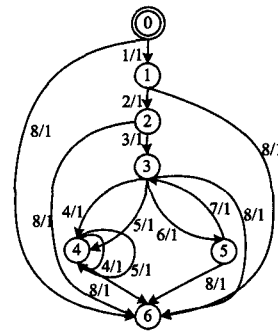


图7 推断得到的POP3协议状态机

对于以上推断结果，需要说明两点：

(1)在设置参数ε时，由于ε表示的是每次生成的测试样本是反例的最大概率，无论采用完全随机还是样本变异的测试样本生成方法，都无法直接确定ε，因此它是假设值也是经验值。在实验过程中还分别将ε取值为0.01和0.03，发现当$\varepsilon=0.03$时，存在因生成的测试样本数过少而无法寻找到反例的情况；而当$\varepsilon=0.01$与$\varepsilon=0.02$时，在输出M_M前均可在r_n次测试样本内找到反例，但是最后一次判定需要的测试样本数增加了一倍，遵从实验成本最小的原则，最终设置$\varepsilon=0.02$。

(2)Shahbaz在文献[11]中已经证明了利用L_M^+算法得到的状态机是最小且完备的，且算法的时间复杂度为关于状态数和最长反例长度的多项表达式。本文所提的L_N^+算法主要对L_M^+算法生成并发送至Oracle的output query和测试样本进行了优化处理，并没有改变L_M^+算法的推断机制，所以可以说明L_N^+算法推断得到的状态机也是最小且完备的，且时间复杂度相同。为了直观地展示状态迁移的主要流程，图6与图7展示的是去除了“dead state”的结果，即不包含在某状态下输入被定义为无效输入报文后转移至的状态。

同时，在同样的测试环境下，对相同的实验样本应用L_M^+算法推断得到的状态机与图6和图7相同，所以，L_N^+算法并没有遗漏状态和转移信息，具备与L_M^+算法同样的推断准确度。

4.2.3 推断效率

L_N^+与L_M^+算法的推断效率主要体现在每次生成候选Mealy机M_M发送至Oracle的ouptut query数目，以及对其进行equivalence query判定时发送至Oracle的测试样本数目。由于在生成测试样本进行equivalence query判定时，存在随机的过程，上述3个效率指标在相同的实验条件下可能不会完全相同。为了避免因偶然因素而导致实验数据不具备说服性的问题，本文的实验数据都取多次实验下的平均值。

首先，表5展示的是L_N^+与L_M^+算法在10次实验中生成的M_M平均数，其中，Cho表示文献[12]中对L_M^+算法的改进方法。需要说明的是，在对Cho方法进行重现时，由于其在L_M^+算法基础上提出的query并行和缓存机制是针对僵尸网络的，而本文采取的实验方式为一对一的C/S方式，因此并没有重现该机制，仅重现了其中对包含自循环的output query的预测机制。但是，L_N^+算法同样适用于对僵尸网络的分析。

表 5　L_N^+ 算法与 Cho 方法生成的 M_M 平均数比较

	Cho	L_N^+	减少比例
SMTP	4.7	3.8	19%
POP3	5.6	4.4	21%

分析表 5 可得，L_N^+ 算法在 SMTP 和 POP3 的协议状态机推断过程中，约平均少生成了一次 M_M，进而会产生更少的输出询问和测试样本。

其次，为了比较当生成相同数量 M_M 时，在推断生成第 n 个 M_M 的过程中（横坐标）L_N^+ 与 L_M^+ 算法发送至 Oracle 的输出询问数目和测试样本数目（纵坐标），我们选择了在 10 次实验中生成相同 M_M 的 3 组实验数据，如图 8 和图 9 所示，其中 SMTP/POP3(N)代表本文提出的 L_N^+ 算法，SMTP/POP3(M)代表 Cho 方法。

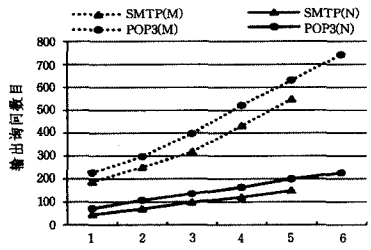


图 8　输出询问(output query)数目比较

分析图 8 可知，在对两种协议的状态机进行推断时，L_N^+ 算法实际发送至 Oracle 的 output query 数目远小于 Cho 方法。由于本文选取的实验对象状态规模较小，只能展示出该规模下的 output query 数目比较情况，但是随着推断的进行，OT 的规模不断扩大，可以看出 Cho 方法的 output query 数目维持着较大的增长趋势，而 L_N+ 算法一直都只有较小的增长，从而可以推断，当状态规模增大时，L_N+ 算法对 output query 数目的优化会更具优势。

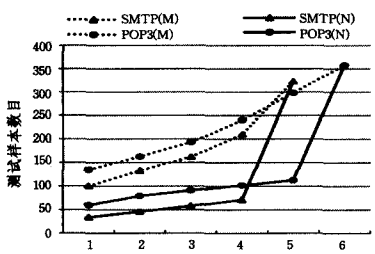


图 9　等价询问(equivalence query)需要的测试样本数目比较

对于图 9，除了进行最后一次等价判定时因一直没有寻找到反例，两种方法均需要 r_n 个测试样本才能近似判定该 M_M 与真实状态机等价，终止推断。在之前的等价判定过程中，L_N^+ 算法中基于正例样本变异的 equivalence query 近似判定算法 APSM 体现了较好的优化效果，寻找到反例需要的测试样本数的增长趋势与 Cho 方法相比有较大幅度的减小。同理可推断，随着状态规模的增大，相比于 Cho 方法，L_N^+ 算法能更好地提升等价判定的效率。

表 6 给出了在 10 次实验过程中，L_N^+ 算法与 Cho 方法整体推断效率的一个量化对比，对比项包含单次实验发送至 Oracle 的 output query 平均总数（N_{oq}）、测试样本平均总数（N_{test}）以及平均推断时间（$Time$）。从表 6 中的平均推断时间可得，相比与 Cho 方法，L_N^+ 算法对 SMTP 和 POP3 的推断效率分别提升了 56%和 58%，体现了较好的优化效率。同时依据对图 8 和图 9 的论述可知，随着状态规模的增大，L_N^+ 算法的优化效果会更加明显。

表 6　L_N^+ 算法与 Cho 方法推断效率的量化比较

	SMTP			POP3		
	Cho	L_N^+	减少比例	Cho	L_N^+	减少比例
N_{oq}（个）	1679	453	73%	2878	748	74%
N_{test}（个）	813	462	43%	1277	701	45%
Time(min)	80	35	56%	127	53	58%

结束语　本文在 L_M^+ 算法的基础上，充分学习捕获的会话样本集，提出了一种基于域知识的协议状态机主动推断算法 L_N^+。通过对两种文本协议的实验测试表明，该算法不仅大幅减少了发向 Oracle 的无效 output query 的数量，而且较大幅度地提升了对 equivalence query 的判定效率。

文中采用的协议逆向分析平台 Netzob 是基于网络流量分析方法的，对获取文本协议报文的格式和语义信息效果较好，但对二进制协议则存在较大局限，限制了对二进制协议状态机的推断。基于执行轨迹的方法更适用于二进制协议分析[21]，可作为进一步研究协议状态机推断的方向。

参 考 文 献

[1] 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
Li W M，Zhang A F，Liu J C，et al. An Automatic Network Protocol Fuzz Testing and Vulnerability Discovering Method [J]. Chinese Journal of Computers，2011，34(2)：242-255

[2] 侯莹，洪征，潘璠，等. 基于模型的 Fuzzing 测试脚本自动化生成[J]. 计算机科学，2013，40(3)：206-209
Hou Y，Hong Z，Pan F，et al. Model Based Automatic Fuzzing Script Generation [J]. Computer Science，2013，38(3)：206-209

[3] 应凌云，杨轶，冯登国，等. 恶意软件网络协议的语法和行为语义分析方法[J]. 软件学报，2011，22(7)：1676-1689
Ying L Y，Yang Y，Feng D G，el al. Syntax and Behavior Semantics Analysis of Network Protocol of Malware[J]. Journal of Software，2011，22(7)：1676-1689

[4] Tridgell. How Samba was written [EB/OL]. 2005-02-04[2014-11-10]. http://samba. org/ftp/tridge/misc/french_cafe. txt

[5] Lang K J. Faster Algorithms for Finding Minimal Consistent DFAs[R]. NEC Research Institute，1999

[6] Oncina J，Garcia P. Inferring regular languages in polynomial update time [J]. Pattern Recognition and Image Analysis，1992，1：49-61

[7] Lang K J，Pearlmutter B A，Price R A. Results of the Abbadingo one DFA learning competition and a new evidence-driven state merging algorithm[M]// Grammatical Inference. Springer Berlin Heidelberg，1998：1-12

[8] Gold E M. Language Identification in the Limit [J]. Information and Control，1967，10(5)：447-474

[9] Dupont P，Lambeau B，Damas C，et al. The QSM algorithm and its application to software behavior model induction [J]. Applied Artificial Intelligence，2008，22(1)：77-115

[10] Angluin D. Learning regular sets from queries and counterexamples [J]. Information and computation，1987，75(2)：87-106

[11] Shahbaz M. Reverse Engineering Enhanced State Models of Black Box Software Components to Support Integration Testing [D]. Laboratoire Informatique de Grenoble，2008

[12] Cho C Y，Shin E C R，Song D. Inference and analysis of formal models of botnet command and control protocols[C]// Proceedings of the 17th ACM Conference on Computer and Communi-

cations Security. ACM,2010:426-439

[13] Caballero J, Poosankam P, Kreibich C, et al. Dispatcher: Enabling active botnet infiltration using automatic protocol reverse-engineering[C]//Proceedings of the 16th ACM Conference on Computer and Communications Security. ACM,2009:621-634

[14] Wikipedia Contributors. Mealy machine [EB/OL]. 2013-11-13 [2014-11-10]. http://zh.wikipedia.org/wiki/Mealy%E6%9C%BA

[15] 肖明明,余顺争. 基于文法推断的协议逆向工程[J]. 计算机研究与发展,2013,50(10):2044-2058
Xiao M M, Yu S Z. Protocol Reverse Engineering Using Grammatical Inference[J]. Journal of Computer Research and Development,2013,50(10):2044-2058

[16] 潘璠,洪征,杜有翔,等. 基于递归聚类的报文结构提取方法[J]. 四川大学学报(工学版),2012,44(6):137-142
Pan F, Hong Z, Du Y X, et al. Recursive Clustering Based Method for Message Structure Extraction [J]. Journal of Sichuan University (Engineering Science Edition),2012,44(6):137-142

[17] Comparetti P M, Wondracek G, Kruegel C, et al. Prospex: Protocol specification extraction[C]//2009 30th IEEE Symposium on Security and Privacy. IEEE,2009:110-125

[18] Bossert G, Guihéry F. Security Evaluation of Communication Protocols in Common Criteria[R]. Paris: ICCC,2012

[19] Postel J. RFC821: Simple Mail Transfer Protocol [S]. IETF, 1982

[20] Myers J, Rose M. RFC1939: Post Office Protocol-Version3[S]. IETF,1996

[21] 潘璠,洪征,周振吉,等. 语义层次的协议格式提取方法[J]. 通信学报,2013,34(10):162-173
Pan F, Hong Z, Zhou Z J, et al. Protocol format extraction at semantic level [J]. Journal on Communications,2013,34(10):162-173

(上接第 223 页)

支,每个合取支包含且仅包含了 m 个元证书,元证书满足门限隶属度 τ 的概率为 β'_{ij},则任意合取支得以满足的概率为 $\prod_{j=1}^{m}\beta'_{ij}$,整个析取式能够满足的概率为 $1-\prod_{i=1}^{n}(1-\prod_{j=1}^{m}\beta'_{ij})$。若所有元证书能够满足的概率均为 β',则证书 s 能够成功披露的概率为 $Prob(s)=1-(1-\beta'^{m})^{n}$。一般地,证书 s 距离元证书的平均距离为 $k(k=0,1,2,\cdots)$,若平均每个证书的协商规则包含了 n 个合取支,每个合取支包含了 m 个平均距离为 $k-1$ 的证书,则 s 得以成功披露的概率为

$$Prob(s_k)=\begin{cases}\beta', & k=0\\ 1-(1-Prob^{m}(s_{k-1}))^{n}, & k=1,2,\cdots\end{cases}$$

类似地,目标资源 R 被成功披露的概率为 $Prob(R)$,故协商成功率为 $Prob(R)$。例如,当 $\beta=0.7$,$\beta'=0.71$,$m=3$,$n=3$,$k=4$ 时,传统的信任协商成功率为 88.7%,而模糊协商的成功率提高到了 95%。可以看出,模糊协商对协商成功率影响非常大。

由于当且仅当找到一个能够披露的根节点的子树时协商可以成功,而双方的协商过程可以看作对这个搜索树的深度优先遍历。若在协商搜索树中目标资源 R 有 n 个子节点,每个子节点在传统的信任协商中成功率均为 $Prob_r$,在模糊信任协商中成功率均为 $Prob_r'$,每个子节点因协商失败造成的效率损失平均为 t_0,则模糊协商比传统的信任协商效率提升了 $t_0\sum_{i=0}^{n}Prob_r'-Prob_r$。由于模糊协商中会根据当前各个证书的模糊隶属度优先披露更有可能成功的证书,因此实际应用中其效率会略高一些。

结束语 自动信任协商为开放网络环境中的用户提供了一种有效而可靠的资源共享方式。针对传统的协商方式存在由于协商规则制定较严格而导致协商成功率和效率有所下降的问题,本文提出了一种基于模糊逻辑的信任协商方案,即协商双方可以通过制定更为合理的信任规则,来尽可能地降低协商的失败率,提高协商效率;同时也简单分析了模糊协商方案下效率的提升。在进一步的工作中,可以针对证书的隶属度做进一步的优化,以提高协商的成功率和效率。

参 考 文 献

[1] Winsborough W H, Seamons K E, Jones V E. Automated trust negotiation[C]//Proceeding of the DARPA Information Survivability Conference and Exposition. 2000:88-10

[2] Seamons K E, Winslett M, Yu T. Limiting the disclosure of access control policies during automated trust negotiation[C]//Network and Distributed System Security Symp (NDSS 2001). Internet Society Press,2001

[3] Yu T, Ma X, Winslett M. PRUNES: An efficient and complete strategy for trust negotiation over the Internet[C]//Proc of the 7th ACM Conf on Computer and Communications Security. 2000:210-219

[4] Winsborough W H, Li N H. Towards practical automated trust negotiation[C]//Proc of the 3rd International Workshop on Policies for Distributed Systems and Networks. Monterey, CA, USA,2002:92-103

[5] Yu T, Winslett M, Seamons K E. Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation[J]. ACM Transactions on Information and System Security,2003,6(1):1-42

[6] Bertino E, Ferrari E, Squicciarini A C. Trust-χ: a peer-to-peer framework for trust establishment[J]. IEEE Transactions on Knowledge and Data Engineering,2004,16(7):827-841

[7] 李建欣,怀进鹏. COTN:基于契约的信任协商系统[J]. 计算机学报,2006,29(8):1290-1300
Li Jian-xin, Huai Jin-peng. COTN: A Contract-Based Trust Negotiation System[J]. Chinese Journal of Computers, 2006, 29(8):1290-1300

[8] Guan S Y, Dong X S, Mei Y D, et al. Towards automated trust negotiation for grids[C]//Proc of the IEEE International Conference on Networking, Sensing and Control. 2008:154-159

[9] 李建欣,怀进鹏,李先贤. 自动信任协商研究[J]. 软件学报,2006,17(1):124-133
Li Jian-xin, Huai Jin-peng, Li Xian-xian. Research on Automated Trust Negotiation[J]. Journal of Software,2006,17(1):124-133

[10] 廖振松,金海,李赤松,等. 自动信任协商及其发展趋势[J]. 软件学报,2006,17(9):1933-1948
Liao Zhen-song, Jin Hai, Li Chi-song, et al. Automated Trust Negotiation and its Development Trend[J]. Journal of Software, 2006,17(9):1933-1948

[11] Zadeh L A. Fuzzy sets[J]. Information and Control, 8(3):338-353