基于改进变异树的工控协议模糊测试用例生成方法*

李文轩[1,2,3,4]，尚文利[2,3,4,5†]，和晓军[1]，陈春雨[2,3,4]，曾 鹏[2,3,4,5]

(1. 沈阳理工大学 自动化与电气工程学院，沈阳 110159；2. 中国科学院沈阳自动化研究所，沈阳 110016；3. 中国科学院机器人与智能制造创新研究院，沈阳 110169；4. 中国科学院网络化控制系统重点实验室，沈阳 110016；5. 中国科学院大学，北京 100049)

摘 要：针对现有应用层工控协议在模糊测试过程中用例冗余度高、测试效率低和随机性强等问题，提出一种基于改进变异树的测试用例生成方法。该方法将协议样本数据序列进行树结构化，同时提取协议规约中字段优先级信息，并利用其有效地控制树中各节点属性值的变异程度，从而达到降低测试成本、提高测试效率以及增大发掘漏洞几率的目的。实验结果表明，该测试用例生成方法对提高协议模糊测试性能具有显著的优化效果和漏洞检测能力。

关键词：工控协议；变异树；模糊测试；测试用例

中图分类号：TP301.6 **文献标志码**：A **文章编号**：1001-3695(2020)12-028-3662-05

doi:10.19734/j.issn.1001-3695.2019.07.0556

Fuzzing test case generation method for industrial control protocol based on improved mutation-tree

Li Wenxuan[1,2,3,4], Shang Wenli[2,3,4,5†], He Xiaojun[1], Chen Chunyu[2,3,4], Zeng Peng[2,3,4,5]

(1. *School of Automation & Electrical Engineering, Shenyang Ligong University, Shenyang* 110159, *China*; 2. *Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang* 110016, *China*; 3. *Institutes for Robotics & Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang* 110169, *China*; 4. *Key Laboratory of Networked Control Systems, Chinese Academy of Sciences, Shenyang* 110016, *China*; 5. *University of Chinese Academy of Sciences, Beijing* 100049, *China*)

Abstract: There are some problems of high redundancy, low test efficiency and strong randomness in the processing of fuzzing test for the existing application layer industrial control protocol, this paper proposed a test case generation method based on improved mutation-tree. The method performed tree structure on the protocol sample data sequence, extracted the field priority information in the protocol specification, and used it to effectively control the degree of variation of the attribute values of each node in the tree, thereby reducing test cost, improving test efficiency, and increasing the probability of exploiting vulnerabilities. The experimental results show that the test case generation method has significant optimization effect and vulnerability detection ability for improving the protocol fuzzy test performance.

Key words: industrial control protocol; mutation-tree; fuzzing test; test case

0 引言

如今人工智能时代背景下的大数据、云计算等信息产业高并发涌现。网络化与信息化的高速发展给生产、生活等方面带来极大便利的同时，随之而来的安全问题却呈现出逐年上升趋势，其严重程度已经上升至国家战略层面，并在逐步蔓延之中[1]。近年来，由于 IT 技术与功能性需求相结合的势态席卷全球，自称封闭、隔离即为安全的工业控制系统难免受到波及，正因为工控系统始终将功能实时性放在首位，并没有将传统网络中的 IT 安全因素加以考虑，盲目地将工控系统网络接入互联网，并且没有采取任何具有针对性的安全防御措施，忽略了传统网络所带来继承性的安全问题[2]。直至 2010 年伊朗核电站震网病毒的爆发，该事件被认为是世界范围内第一个造成物理伤害的网络攻击，其所造成的严重后果已经威胁到整个工业控制系统的资产以及个人的生命财产安全，可以说没有工控系统网络安全就没有工控系统的生产安全，因此工控系统安全引起了许多攻击者及研究人员的密切关注[3]。

众所周知，工业控制系统由大量的嵌入式以及网络通信设备所构成，但就工控系统内部网络而言，其总线网络内嵌工控网络通信专有协议，而外部连接网络仍为基于 TCP/IP 协议的传统以太网结构[4]。由于协议设计者仅从可用性角度出发，并未考虑其安全需求，同时又借助 TCP/IP 建立工控协议栈，难免会将传统 IT 网络存在的漏洞引至工控网络内部，因此工控系统面临着巨大的安全挑战[5]。工控系统之所以遭受攻击，其原因不外乎是系统及通信规约存在着安全缺陷，但基于诸多因素的考虑，工控通信协议中的漏洞并不像传统 IT 那样利用相关补丁便能够实时地进行漏洞修复，因此针对工控通信协议进行有效的漏洞挖掘是发现并剔除工控脆弱性的必要手段。

模糊测试作为一种有效检测安全漏洞的黑盒测试方法[6]，其主要针对软件领域中存在的安全缺陷加以测试，许多大中型软件开发商将其作为保证软件安全质量中测试流程的一部分，尽可能地达到发现软件漏洞缺陷的目的[7]。随着模

收稿日期：2019-07-23；**修回日期**：2019-09-17 **基金项目**：国家重点研发计划项目(2018YFB2004200)；中科院战略性先导科技专项项目(XDC02020200)；国家自然科学基金资助项目(61773368)

作者简介：李文轩(1994-)，男，黑龙江双鸭山人，硕士研究生，主要研究方向为工业控制系统信息安全、漏洞挖掘等；尚文利(1974-)，男(通信作者)，黑龙江北安人，研究员，博导，博士，主要研究方向为工业控制系统信息安全、计算机智能与机器学习等(shangwl@sia.cn)；和晓军(1968-)，女，辽宁沈阳人，副教授，硕士，主要研究方向为数据仓库和数据挖掘、信息检索和搜索引擎等；陈春雨(1992-)，男，黑龙江哈尔滨人，助理研究员，硕士，主要研究方向为信息安全、人工智能等；曾鹏(1976-)，男，研究员，博导，博士，主要研究方向为工业无线传感、智能电网等.

糊测试的不断发展,该方法进一步沿用至网络协议领域,由于该方法可以检测到某些高风险漏洞[8],所以备受工控安全领域中一些研究人员甚至攻击者的青睐,同时针对一些协议如 Modbus TCP、DNP3 等设计且实现了具有针对性的模糊测试方案,取得了不错的成果[9]。很大程度上传统的模糊测试方法过于依赖于测试人员现有的经验及技术,某些特定情况下仍存在很多不足,如测试效率低、测试用例高度冗余、测试覆盖度低以及测试不充分等问题。

伴随着工控领域安全问题的愈演愈烈,近些年安全研究者提出一些基于模型的 fuzzing 测试方法,例如 Wu 等人[10-12]利用树结构对协议进行建模,通过提取遗传变异因子实现单、多样本组合测试,有效地解决了过分依赖测试样本数据的问题;同时采取多维输入结合遗传算子的变异方法,利用算子直接触发不安全目标函数从而提高模糊测试发掘脆弱性的能力。张亚丰等人[13]从协议语义角度出发,构建改进范式语法变异树描述工控协议,并提出相应的测试用例生成算法,对测试用例覆盖度及测试效率均有一定的提高。刘海龙等人[14]提出基于变异树模型的测试方法,利用预先设计变异因子与树模型相结合,有效地解决了变异因子作用后的测试用例效果较差引起的连锁反应,同时也解决了由于前期静态分析错误引发的漏报问题并提高了测试效率。刘静静等人[15]利用分类树结合贪婪算法的思想提出一种测试用例生成方法,利用 FTP 分析并验证了该方法的可行性与有效性。Ma 等人[16]采用启发算子与分类树结合的思想,同时引入笛卡尔积进行协议中逐字段变异,使生成的模糊测试数据更加细粒度,进而大大提高测试用例的质量。由于考虑到工控协议模糊测试中变异算子效率低的问题。Dong 等人[17]在刘海龙等人所设计的 FTSG 模型基础上提出一种改进的变异树优化模型及剪枝算法,引入估值准则进行剪枝优化处理,同时针对每个节点进行标准评估,测试结果表明该方案比 FTSG 模型效率更高,且测试数据量为其 50 倍。Zhang 等人[18]提出一种基于深度学习生成对抗神经网络的 KWP2000 协议漏洞挖掘方法,利用神经网络训练生成测试用例数据并执行模糊测试,实验结果表明该方法检测出超长字符错误、编码错误等漏洞,并且提高了测试效率和安全性。Li 等人[19]提出一种基于 WGAN 生成模糊数据的方法,该方法不需要定义输入数据格式,利用该模型可以短时间训练产生测试数据,具有计算量小、实用性强等特点。

综上所述,尽管采用深度学习模型解决测试数据具有一定的优势,而从模型解释度来看利用树模型构建协议序列模型可以更直观且有效地表达网络协议中存在的连续性或嵌套关系,但整体上由于测试用例设计不当所导致测试环节中高冗余、低覆盖及低效率等问题仍然存在,并且现有大多数方法均采用基于变异的方式构造测试用例,却忽略了基于生成的方式在用例构造时的比重,同时也没有考虑到协议字段间所存在的约束关系对变异程度进行优化调整,最后造成测试过程存在着盲目性。因此本文在基于变异与生成的基础上,针对工控通信系统中主流的应用层协议提出基于改进协议变异树与优先级修剪算法相结合的测试用例优化生成方法,进一步解决测试用例冗余所导致测试效率低等问题。

1　基于树模型的工控协议描述

本文基于现有的工控协议树构建基础,将工控协议树模型结构自上而下分为三部分,即目标协议、语义可分和字节可分,其结构如图 1 所示。图 1 中 P 对应待测目标协议,S_n 为协议报文中语义可分割域或字段,B_n 表示将报文序列拆分为字节可分割部分。因此,工控协议树模型可定义为以待测协议作为协议树入口节点,由上至下将协议序列分割为语义可分和字节可分两部分,即中间节点和叶子节点,并根据协议规约明确可变异节点,同时剔除不可变异节点,从而降低无效变异作用域,为构造变异树和测试用例生成奠定基础。

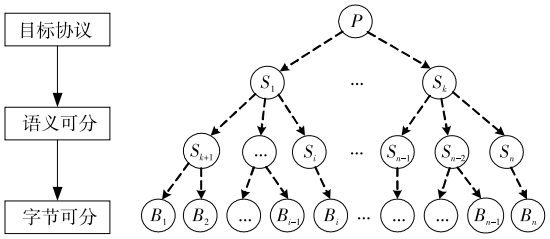

图 1　工控协议树结构描述

Fig. 1　Industrial control protocol tree structure description

由于本文主要针对应用层协议进行模糊测试,总体上可将协议本身划分为各个协议单元,有的称其为通信对象或者数据单元,但从语义上主要分为源地址、目的地址、命令 ID、设备 ID、协议标志位、数据长度、功能码、状态标志位、数据字段、校验字段以及其他辅助标志位等部分。不同应用层协议由不同的部分组成,但不同协议之间的控制字段在功能上有着异曲同工之处,比如 Ethernet/IP 中的 CIP 数据包的命令字段、ModbusTCP 中的功能码以及 CANOPEN 协议中的 FunctionCode 都具备着相应的控制功能,并且不同的字段值表示不同的读写操作,同时各协议报文序列中也都具有数据长度、数据字段和状态标志字段,尤其是通过主站回复从站的响应状态码可判断连接状态和请求数据包的合法性等。通过上述分析可知,工控应用层协议具有近似统一的协议单元,可利用协议树模型对其进行形式化和结构性描述,并完全能够表示协议序列中各字段的层次关系。

在通信过程中最重要的是各个协议中字段之间的关联性和主从通信时字段优先级的校验顺序,显然这需要根据不同的协议规约进行具体分析。一般情况下,通过特定协议号判别是否为该协议报文序列,随后根据功能码或者命令字段对其后续字段进行字段值及字段间嵌套关系的合法性验证,接着根据服务端回复响应报文序列中的状态码判断请求报文数据是否合法。因此本文将根据协议规约和测试过程中响应码对生成的变异树进行局部剪枝处理,进而构造出高效的测试用例。

2　基于变异树模型的工控协议测试用例方法

根据上述工控协议树模型中的结构定义,首先应构建初始协议样本树并提取变异因子制定变异函数体,其中每个变异体施加在初始协议样本树时,可作用于单个或多个可变节点及其子节点;其次利用协议字段间的优先级关系对初始变异后无效的节点进行预剪枝处理,主要是为了限制特殊定值节点的变异,在构造之前降低冗余度,最终得到优化后的变异树进而生成测试用例。

2.1　建立工控变异树模型

本文通过建立五元组改进模型对工控变异树进行形式化描述,其具体形式为

$$\mathrm{ICPT}=\{\langle SD,TS,\mathrm{priCondtion},\mathrm{mutationFunc},\mathrm{testResult}\rangle\}$$

该模型各部分定义如下:

$SD=\{sd_1,sd_2,\cdots,sd_i,\cdots,sd_n\}$,$SD$ 为初始协议样本数据。其中:sd_i 为输入协议的字节或字段样本数据,$1\leqslant i\leqslant n$。

$TS=\{\langle \mathrm{nodes},\mathrm{pos},\mathrm{afflication}\rangle\}$,$TS$ 表示树结构中的关系集合。其中 $\mathrm{nodes}=\{\langle L,NL\rangle\}$ 为树中节点集,$L=\{l_1,l_2,\cdots,l_i,\cdots,l_n\}$ 表示协议树中叶节点集合,且 l_i 为工控网络协议中不能分割的语义节点单元即字节可分,$1\leqslant i\leqslant n$;$NL=\{Nl_1,Nl_2,\cdots,Nl_i,\cdots,Nl_n\}$ 表示非叶子节点集合,Nl_i 为协议中可分割的语义单元,$1\leqslant i\leqslant n$;pos 表示节点的位置;afflication 表示节点间的从属关系,如父子关系、兄弟关系等。

PriCondtion 为字段优先级约束条件,表示根据协议字段关

系及属性值范围等约束条件控制树节点值变异方向。

MutationFunc 为作用于树节点的变异函数体集合，MutationFunc = $\{func_1, func_2, \cdots, func_i, \cdots, func_n\}$。其中，$func_i$ 表示第 i 个变异函数体，$1 \leqslant i \leqslant n$。

TreeResult = $\{\langle MT, TestCases\rangle\}$。其中 MT 为变异树集合，mt_i 为集合中任意一棵变异树；TestCases 为测试用例集合，$testcase_i$ 为待执行测试用例序列，且 $mt_i \xrightarrow{traverse} testcase_i$，$1 \leqslant i \leqslant n$。

2.2 初始化工控协议样本树

针对工控协议进行模糊测试需要根据协议规约并利用捕获报文数据进行协议格式分析，而基于树结构的层次解析恰恰能够自上而下且完整地表达协议字段间的嵌套关系，因此本文首先对待测协议建立协议样本树结构，其建立过程如算法 1 所示。

算法 1 协议样本树生成算法

输入：样本数据集 $SD = \{sd_1; sd_2; \cdots; sd_i; \cdots; sd_n\}$；nodes = { }；
树结构关系集 $TS = \{\langle nodes, pos, affliction\rangle\}$。
输出：协议样本树集 $ST = \{st_1, st_2, \cdots, st_i, \cdots, st_n\}^T$。

```
a)  for each SD in SD do
b)      if SD≠∅ then
c)      while(ByteSplit or SemanticSplit)
d)          partition(SD)←ProtocolRule;
e)          Nodes.join(L,NL);
f)      end while
g)      end if
h)      for each node in Nodes do
i)          st_i←GenerateTree(Nodes,TS);
j)          clear(Nodes);
k)      end for
l)  end for
```

将输入的协议样本数据集表示为向量集合的形式，每一向量视做一个协议样本输入序列 SD，根据 ProtocolRule 对 SD 进行语义分割或字节划分并加入到 Nodes 集合中，利用 GenerateTree 函数建立协议样本树，目的在于梳理工控协议字段间的依赖关系，以至于在后续工作中无遗漏地对某一或多个节点执行变异操作，从而能够保证在测试过程中测试用例全覆盖。

2.3 约束优先级策略的变异树生成

在生成变异树之前，为了尽可能地降低变异复杂度，本文需要对样本树进行预剪枝处理，目的在于限制初始协议树中特定节点及其子树节点属性值的变异操作，其主要从协议规约和捕获的协议样本数据中提取不可变异协议段，如 ModbusTCP 固定的协议号 0x00，甚至某些协议的固定端口号以及相对应的 IP 地址等都被归纳为不可变异因素，因此建立不可变字段集合 NonMutatField，并对协议树中可变异和不可变异节点进行初始状态标记，其具体过程如下所示。

算法 2 协议树预剪枝算法

输入：NonMutatField = $\{Nm_1, Nm_2, \cdots, Nm_n\}$；
协议样本树集 $ST = \{st_1, st_2, \cdots, st_i, \cdots, st_n\}^T$；
树结构关系集 $TS = \{\langle Nodes, Pos, Affliction\rangle\}$。
输出：带有变异标志状态的协议树 StatusTree = { }。

```
for st in ST do
    node←st.root;
    MarkStatus(node,TS){
        temp←count(node.hasChild());
        for tmp in temp do;
            if(node.childNode[tmp]∈NonMutatField)
                tmpNode.status&&tmpNode.substree.status=1;
            else
                tmpNode.status=0;
                MarkStatus(tmpNode.subNode,TS);
        end for
    }
end for
```

上述过程在初始协议树的基础上确定了待变异节点的范围，从根节点循环遍历子节点并标记变异状态位，若某一节点的属性值在 NonMutatField 集合中，则状态位置 1，否则置 0。这样利用预剪枝很大程度上缩减了变异范围，间接地降低了无效用例的生成数量。为了进一步控制树中各节点的变异程度，将带有变异标志协议树中的每个节点都视为一个潜在的待变异单元，其有助于构成以该节点为首的测试用例序列子集，因此本文构造变异函数体对节点属性值施加变异策略，利用该策略控制协议样本树中局部甚至全局节点的变异趋势。根据国家信息安全漏洞共享平台 CVND[20] 所示的工控系统主要攻击漏洞成因及传统模糊测试所涉及的变异因子进行函数体构造，其变异函数如表 1 所示。显然，可变异状态位为 0 的协议树节点只能采用零变异操作，而状态位为 1 的节点可根据约束条件选取适当的变异操作，但利用这些变异函数体直接操作这些树节点往往并不能达到令人满意的测试效果，需要增加一些限制条件降低随机变异所导致的盲目性。

表 1 部分变异函数体描述

Tab. 1 Description of partial variogram bodys

变异类型	变异函数体	变异说明
零变异	$node_i = I(node_i), i \in [1, n]$	返回自身节点值
特殊字符	{%, ~, #, %d, %f, /, .. /, ...}	替换或插入原始字段
边界值	{min ± 1, max ± 1, min, max, 0, ± 1, ⋯}	协议字段范围临界值
不定长字符串	$node_i$.fill(len * 'character'), $len \in Z^*$	填充字段变异
操作节点	{Del($node_i$), Modify($node_i$), Add($node_i$), ...}	部分节点的增删改
比特位翻转	ReverseBits($node_i$ | from, to}), from, to ∈ [0, Bits($node_i$)]	指定节点位区间翻转

通过深度解析协议规约不难看出，协议字段间存在着优先级校验顺序，但一些安全测试人员并未注意到这一细节。而在测试过程中，字段优先级却具有重要的作用，协议字段及其次序的正确性决定着能否正常接收并处理客户端发送的请求，即次级字段受到前一级字段的影响，若第一个字段为合法值，则次级字段将起到关键作用。综上所述，由于协议字段优先级条件的存在，致使在主从通信的过程中可根据请求一响应的状态码调整测试用例中字段的构造，从而大范围地降低无效用例的产生。

就工控系统中所有应用层协议而言，建立并保持应用层协议连接和请求报文的正确接收是整个 Fuzzing 协议测试过程中的前提基础，因此本文基于 ModbusTCP 分析字段优先级顺序在协议通信过程中所起到的作用。如图 2 所示，分别对 MBAP 报文头和 PDU 两部分进行校验。首先，协议类型也就是请求报文中的协议标志符是否为 0x00 决定着事务处理能否成功建立，因此协议类型字段对应树中节点及其子节点设置为不可变异状态；其次，事务处理 ID 逐次加 1，并验证其是否在事务处理属性值的范围内；最后验证功能码是否有效。

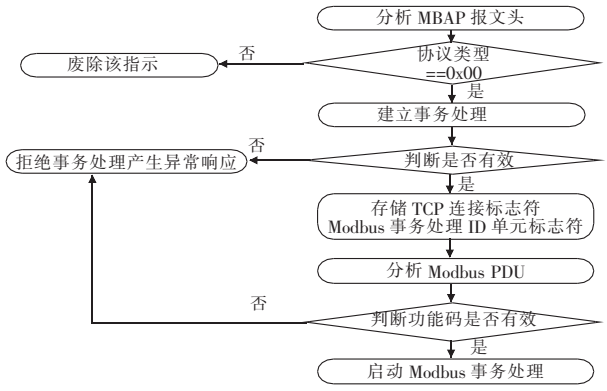

图 2 基于字段优先级的协议校验顺序

Fig. 2 Check order based on protocol field priority

由此可以看出，对工控协议进行 fuzzing 测试的本质是基于不同功能码或命令码所代表的特定功能进行测试，但不同工控协议的验证次序各不相同，具体情况需要参照协议规约。响应报文中的状态码对解决无效测试用例起到较好的局部修剪

作用,在变异作用域内利用变异函数体对协议树采取变异操作并生成初始测试用例,因此本文基于初始变异树及其测试用例生成的具体算法过程如下所示。

算法 3　初始变异树及其测试用例生成

输入:带有变异标志状态的协议树 StatusTree = { }。

输出:初始变异树集 IMT = { };初始测试用例集 ITC = { }。

```
for st in StatusTree do
    for node in st. nodes do
        for each function enumerate( Functions) do
            if( node. status == 0)
                node←Mutation( node,ZeroMutation) ;
            else
                node←Mutation( node,MutateFuction) ;
            end if
        end for
    end for
    imt←update( st) ;
    IMT. append( imt) ;
    itc←imt. traverse( ByteNodes) ;
    ITC. append( itc) ;
end for
```

上述伪代码用于初始变异树的确定及初始测试用例的生成,由于在分析过程中根据 ModbusTCP 自身协议规约的要求,ProtocolID 字段为不可变异字段,所以只能对该节点及其子节点采取零变异操作,而对于其他节点来说,可以通过枚举变异函数对节点施加变异作用,并将得到的初始变异树结果及对应测试用例记录到 IMC、ITC 待测集合中。为了进一步优化测试性能,本文设计并实现了节点后剪枝算法,需要对初始生成的测试用例执行预测试,通过对测试过程中异常响应报文的分析进而回溯变异树修改可变异节点或子节点,从而满足协议通信的需求。下面通过写多个线圈的功能码 0x0F 来具体说明剪枝流程,其算法的具体过程如下所示。

算法 4　变异树后剪枝算法

输入:初始测试用例集 ITC。

输出:变异树集合 *MT* = { };有效测试用例集合 *TC* = { }。

```
for itc in ITC do
    target←execute( itc) ;
    if itc. responCode == Invalid then
        InvalidNum ++ ;
    TC. append( itc) ;
    while InvalidNum > Threshold do
        if FunctionCode == false && ExceptionCode == 0x81 then
            / * strategy_1 * /
        else if OutputValue == false && ExceptionCode == 0x83 then
            / * strategy_2 * /
        else if StartAddr == false &&( StartAddr + OutputValue) ==
false && ExceptionCode == 0x82 then
            / * strategy_3 * /
        else if MultiOutput == false && ExceptionCode == 0x84 then
            continue;
        end if
        target←execute( itc') ;
        if itc'. responCode == Valid then
            TC. append( itc') ;
            InvalidNum--;
            break;
        end if
    end while
end for
```

在剪枝过程中 threshold 为执行测试用例返回报文响应状态为无效的设定阈值即循序剪枝的批量值,根据响应状态的异常程度人为设定初始阈值并执行测试流程,并将正常响应的测试用例加入到集合 *TC* 中,其中当功能码 FunctionCode 回显状态为异常时,strategy_1 则计算当前异常用例的字段属性值索引至相应子树节点,并对可变节点值在取值范围内进行变量修改;若输出值 OutputValue 为非法值时,strategy_2 则根据功能码 0x0F 针对变异树中寄存器输出数量节点及其子节点,在阈值 0x0001 ~ 0x07B0 内该节点属性值进行计算调整,并按照式(1)调整该节点下的可变节点属性值,其中 InputNum 为请求报文中的输入比特位数量,N^* 为报文中的字节数值。

$$N^*(\text{InputNum}) = \begin{cases} 1 + \text{InputNum}/8 & \text{mod}(\text{InputNum}) \neq 0 \\ \text{InputNum}/8 & \text{mod}(\text{InputNum}) = 0 \end{cases} \quad (1)$$

其中:strategy_3 实质上是对 strategy_2 的扩充,同时在 OutputValue 的基础上增加 StartAddr 的范围值进行约束,使得节点属性值必须满足 0x0000 ~ 0xFFFF 这一范围,接下来的步骤按照 strategy_2 执行构造;对于抛出异常码 0x84 的测试用例不再对其进行修剪处理;最后发送修改后的用例执行测试,若该用例能正常响应则加入到集合 *TC* 中。

显然,针对不同的功能码进行生成测试用例时,变异树中节点属性值的约束标准各不相同,因此对于不同的应用层协议来说,需要对待测协议规约进行深度解析,比如 CIP 中请求数据包中的状态码以及选项字段值都为 0,其中若字段值不为 0,则表示该数据包处于被丢弃状态,所以应对这两个字段采取零变异策略;ModbusTCP 基于不同功能码进行测试,而 CIP、CANOPEN 等应用层协议是基于命令字段,两者具有相同的功能;当然也会根据请求内容返回不同的状态码。因此该方法利用协议字段优先级可以降低测试冗余,提高测试用例的质量并针对应用层协议有一定的普适性。

3　实验结果与分析

3.1　实验设计

本文针对不同版本的 Modbus Slave 进行模拟仿真测试,从而对该方法进行有效性验证。首先结合协议规约并利用协议样本数据构建初始协议样本树,实际上通过树模型结构对 ModbusTCP 请求报文序列中的协议字段及其字节进行自动拆分。通过整理目前存在的 ModbusTCP 协议漏洞,提取并设计单变异和组合变异因子,进而基于变异树的测试用例变异模块 MutateTree 实现测试用例的组合生成,并编写执行测试用例脚本实现自动化测试。对于测试期间的响应状况,本文则通过 Wireshark 工具进行旁路监听,并将所捕获的 pcap 文件分析测试用例的有效程度及执行效率、导致异常的用例和异常响应信息,其具体测试执行过程如图 3 所示。

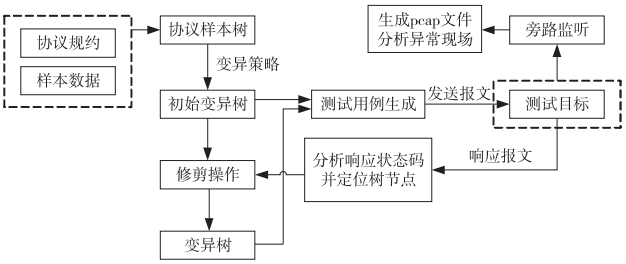

图 3　基于变异树的测试执行流程

Fig. 3　Flow chart of test execution based on mutation-tree

3.2　结果分析

本文对待测目标协议的测试执行时间设定为 2 h,同时为了降低随机性对实验结果所带来的影响,采取测试 5 次取平均值的方式进行最终的结果分析。通过对测试过程中捕获报文所生成的 pcap 文件进行统计分析可得出如表 2 所示的测试结果。

表 2　变异树与 peach 的测试结果对比

Tab. 2　Comparison of test results between mutation-tree and peach

测试方案	ModbusTCP 请求数量	ModbusTCP 正常响应数量	ModbusTCP 异常响应数量	测试执行时间/h
变异树	32 135	20 647	2 319	2
peach	54 293	8 250	5 513	2

从上述实验结果中报文请求—响应的正常交互数据可以看出,基于变异树测试方案能够较好地降低测试用例的冗余

度,通过两者的异常响应数量对比可知,该方案能够大大提高测试用例的质量,降低了测试开销。同时本文通过对文献[21]中所采用的测试效率指标进行更新,并用于评价该方案的测试效率,其评价表达式为

$$\alpha = \frac{\sum \text{validMsg}_i}{\sum \text{Msg}_i} \tag{2}$$

其中:α 表示测试报文的有效执行效率;i 表示测试时间段;分子表示第 i 段时间内发送的有效报文数量;分母表示第 i 段时间内发送测试报文数量的总数。

图 4 为有效测试用例及测试效率与时间的关系。

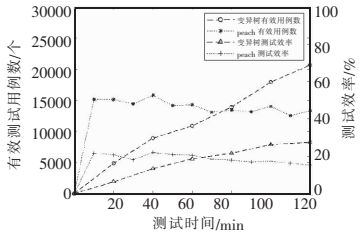

图 4 有效测试用例及测试效率与时间的关系

Fig. 4 Relationship of effective test cases and test efficiency versus time

从图 4 中可看出,该变异树方案有效用例数以及测试效率均要高于 peach 模糊器,这是由于 peach 变异报文的过程中随机程度过高,从而导致测试执行效率过低。该方案利用协议中优先级字段对变异过程中引导节点进行优化剪枝,尽可能降低测试过程中的盲目性,并提高测试用例质量和测试效率;在漏洞检测方面,针对 Modbus Slave V4.3 和 V7.0 两个版本进行实验测试,peach 测试结果中并未发现异常漏洞,而本文采用的方法却检测出两个漏洞,其中 V7.0 版本中存在超长字符串填充所造成的缓冲区溢出漏洞,另外 V4.3 版本存在着非法写入的未知漏洞,在未经允许情况下能够利用同一传输标志符将数值写入寄存器。综上所述,本文所采用的测试用例生成方法能够很大程度上优化测试效果,并且具备检测漏洞的能力。

4 结束语

本文针对目前工控应用层协议模糊测试过程中存在测试用例冗余度以及测试效率低等问题,提出基于改进变异树的测试用例优化方案,并利用协议规约中的字段优先级关系有效地控制协议树中各节点属性值的变异程度,其变异程度严重影响到无效测试用例的数量和发现漏洞的几率,同时合理地选取协议样本数据对提高测试用例质量也具有积极的作用。实验结果表明,利用该方法可以生成高质量的测试用例,并提高模糊测试的执行效率,同时也增大了触发异常漏洞的几率。

参考文献:

[1] Xia Zhihua, Wang Xinhui, Zhang Liangao, *et al*. A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing[J]. IEEE Trans on Information Forensics & Security, 2017, 11(11): 2594-2608.

[2] Fu Zhangjie, Huang Fengxiao, Ren Kui, *et al*. Privacy-preserving smart semantic search based on conceptual graphs over encrypted outsourced data[J]. IEEE Trans on Information Forensics and Security, 2017, 12(8): 1874-1884.

[3] Knowles W, Prince D, Hutchison D. A survey of cyber security management in industrial control systems[J]. International Journal of Critical Infrastructure Protection, 2015, 9(6): 52-80.

[4] Bill M, Dale R. A survey SCADA of and critical infrastructure incidents [C]//Proc of the 1st Annual Conference on Research in Information Technology. New York: ACM Press, 2012: 51-60.

[5] 陶耀东,李宁,曾广圣. 工业控制系统安全综述[J]. 计算机工程与应用, 2016, 52(13): 8-18. (Tao Yaodong, Li Ning, Zeng Guangsheng. Review of industrial control systems security[J]. Computer Engineering and Applications, 2016, 52(13): 8-18.)

[6] Kargén U, Shahmehri N. Turning programs against each other: high coverage fuzz-testing using binary-code mutation and dynamic slicing [C]//Proc of the 10th Joint Meeting on Foundations of Software Engineering. New York: ACM Press, 2015: 782-792.

[7] Sunjin K, Taeshik S. Field classification-based novel fuzzing case generation for ICS protocols[J]. The Journal of Supercomputing, 2017, 74(9): 4434-4450.

[8] Voyiatzis A G, Katsigiannis K, Koubias S. A Modbus/TCP fuzzer for testing internetworked industrial systems [C]//Proc of the 20th IEEE Conference on Emerging Technologies & Factory Automation. Piscataway, NJ: IEEE Press, 2015: 1-6.

[9] Adam C J, Sergey B. Bolt-on security extensions for industrial control system protocols: a case study of DNP3 SAv5[J]. IEEE Security & Privacy, 2015, 13(3): 74-79.

[10] Wu Zhiyong, Sun Lechang, Tang Heping, *et al*. A fuzzing technology with a data sample combination[C]//Proc of the 2nd International Conference on Computer Engineering and Technology. Piscataway, NJ: IEEE Press, 2010: 140- 145.

[11] Wu Zhiyong, Zhu Xueyong, Atwood J W. A fuzzing technology using multiple data samples combination[C]//Proc of International Workshop on Computer Science for Environmental Engineering and EcoInformatics. Berlin: Springer, 2011: 1-11.

[12] Wu Zhiyong, Atwood J W, Zhu Xueyong. A new fuzzing technique for software vulnerability mining[J]. Journal of Communication and Computer, 2011(8): 88-95.

[13] 张亚丰,洪征,吴礼发,等. 基于范式语法的工控协议 fuzzing 测试技术[J]. 计算机应用研究, 2016, 33(8): 2433- 2439. (Zhang Yafeng, Hong Zheng, Wu Lifa, *et al*. Form-syntax based Fuzzing method for industrial control protocols[J]. Application Research of Computers, 2016, 33(8): 2433-2439.)

[14] 刘海龙,汪永益,潘祖烈. 基于变异树的 fuzzing 技术研究[J]. 计算机工程与设计, 2011, 32(11): 3618-3621, 3632. (Liu Hailong, Wang Yongyi, Pan Zulie. Research on fuzzing technology based on tree-mutator[J]. Computer Engineering and Design, 2011, 32(11): 3618-3621, 3632.)

[15] 刘静静,袁耀东. 基于启发式搜索和分类树的网络协议模糊测试用例生成方法研究[J]. 现代电子技术, 2016, 39(21): 36-39, 43. (Liu Jingjing, Yuan Yaodong. Research on network protocol fuzzy test case generation method based on heuristic search and classification tree[J]. Modern Electronics Technique, 2016, 39(21): 36-39, 43.)

[16] Ma Rui, Ji Wendong, Hu Changzhen, *et al*. Fuzz testing data generation for network protocol using classification tree[C]//Proc of Communications Security Conference. Piscataway, NJ: IEEE Press, 2014: 1-5.

[17] Dong Guofang, Sun Pu, Shi Wenbo, *et al*. A novel valuation pruning optimization fuzzing test model based on mutation tree for industrial control systems[J]. Applied Soft Computing, 2018, 70(9): 896-902.

[18] Zhang Chengbin, Zhao Hui, Cao Zongyu. The vulnerability mining method for KWP2000 protocol based on deep learning and fuzzing [J]. Journal of Shandong University: Engineering Science, 2019, 49(2): 17-22.

[19] Li Zhihui, Zhao Hui, Shi Jianqi, *et al*. An intelligent fuzzing data generation method based on deep adversarial learning[J]. IEEE Access, 2019, 7: 49327-49340.

[20] National Internet Emergency Center. China national vulnerability database [EB/OL]. [2018-09-26]. http://ics.cnvd.org.cn/.

[21] 张洪泽,洪征,周胜利,等. 基于协议状态机遍历的模糊测试优化方法[J]. 计算机工程与应用, 2020, 56(4): 82-91. (Zhang Hongze, Hong Zheng, Zhou Shengli, *et al*. A fuzzing optimization method based on protocol state migration traversal[J]. Computer Engineering and Applications, 2020, 56(4): 82-91.)

[22] 刘龙霞,吴军华. 基于分类树和贪心算法的测试数据自动生成方法[J]. 计算机工程与设计, 2011, 32(8): 2734-2736, 2820. (Liu Longxia, Wu Junhua. Automated test data generation method based on classification-tree and greedy algorithm[J]. Computer Engineering and Design, 2011, 32(8): 2734-2736, 2820.)