计算机科学
COMPUTER SCIENCE

PGNFuzz:基于指针生成网络的工业控制协议模糊测试框架

王田原，武淑红，李兆基，辛昊光，李璇，陈永乐

引用本文

王田原，武淑红，李兆基，辛昊光，李璇，陈永乐. PGNFuzz:基于指针生成网络的工业控制协议模糊测试框架[J]. 计算机科学, 2022, 49(10): 310-318.

WANG Tian-yuan, WU Shu-hong, LI Zhao-ji, XIN Hao-guang, LI Xuan, CHEN Yong-le. PGNFuzz:Pointer Generation Network Based Fuzzing Framework for Industry Control Protocols[J]. Computer Science, 2022, 49(10): 310-318.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.210700248

PGNFuzz:基于指针生成网络的工业控制协议模糊测试框架

王田原　武淑红　李兆基　辛昊光　李　璇　陈永乐
太原理工大学信息与计算机学院　山西 晋中 030600
(wty971212@126.com)

摘　要　工业安全问题一直是重要而紧迫的全球性问题,工控协议被广泛应用于工业控制系统(Industrial Control System,ICS)组件之间的通信,其安全性关系到整个系统的安全稳定运行,迫切需要保证所有工控协议的安全。网络协议模糊测试对保证ICS的安全性和可靠性起着重要的作用,传统的模糊测试方法提高了工控协议的安全性,其中许多方法具有实际应用价值。然而,传统的模糊测试方法严重依赖于工控协议的规范,使得测试过程昂贵、耗时、麻烦和枯燥,如果规范不存在,任务就很难进行。因此,文中提出了一种基于指针生成网络(Pointer-Generator Networks,PGN)的智能且自动的协议模糊测试方法,并给出了一系列的性能指标。在此基础之上,设计了一个自动化智能应用模糊测试框架PGNFuzz,可用于各种工业控制协议。采用Modbus和EtherCAT等几种典型的工控协议对该框架的有效性和效率进行测试,实验结果表明,该方法在便捷性、有效性和效率方面均优于其他通用型模糊器(General Purpose Fuzzer,GPF)和其他基于深度学习的模糊测试方法。

关键词:自动化漏洞挖掘;模糊测试;工业控制协议;工业安全;深度学习;指针生成网络
中图法分类号　TP273;TP309

PGNFuzz:Pointer Generation Network Based Fuzzing Framework for Industry Control Protocols

WANG Tian-yuan,WU Shu-hong,LI Zhao-ji,XIN Hao-guang,LI Xuan and CHEN Yong-le
College of Information and Computer Science,Taiyuan University of Technology,Jinzhong,Shanxi 030600,China

Abstract　Industrial security issues have always been an important and urgent issue globally. Industrial control protocols are widely used in the communication between industrial control system(ICS) components. Their security is related to the safe and stable operation of the entire system,and there is an urgent need to ensure the security of all industrial control protocols. The network protocol fuzzing plays an important role in ensuring the security and reliability of ICS. Traditional fuzzing methods can improve the security testing of industrial control protocols,and many of which have practical applications. However,most traditional fuzzing methods rely heavily on specifications of industrial control protocols,making the test process costly,time-consuming,cumbersome and boring. If the norm does not exist,the task is difficult to carry out. This paper proposes an intelligent and automatic protocol fuzzing method based on pointer-generation networks(PGN),and gives a series of performance indicators. On the basis of this method,an intelligent and automatic fuzzing framework based on PGNFuzz for application is designed,which can be used for various industrial control protocols. Several typical industrial control protocols such as Modbus and EtherCAT are used to test the validity and efficiency of our framework. Experiment results show that our method is superior to other general purpose fuzzers(GPF) and other deep learning based fuzzing methods in terms of convenience,effectiveness and efficiency.

Keywords　Automatic vulnerability mining,Fuzzing,Industrial control protocols,Industrial security,Deep learning,Pointer-generation networks

1　引言

随着信息技术和通信技术[1]的逐步融合,当今的工业控制系统正在从自我封闭的信息孤岛迅速转变为应用广泛的互联架构,这种转变会大大增加工业控制网络被攻击的风险[2]。全球每年针对ICS的安全事件都呈现出急剧上升的趋势,安全漏洞已成为工业信息安全受到威胁的根本原因,尤其是2017年5月爆发的WannaCry勒索病毒,其利用SMB协议中的漏洞对100多个国家和地区超过10万台电脑进行了勒索攻击,造成了高达80亿美元的损失,给金融、能源、医疗等众多行业造成了管理的重大危机和巨大损失。工控协议作为工控系统各部分之间沟通的桥梁,促进了行业信息化建设并提高了生产和管理效率,其安全性关系到整个系统的安全稳定运行。

自20世纪90年代初引入模糊测试技术以来[3],模糊测试在网络协议领域一直有着广泛的应用,并开发了许多工具。

到稿日期:2021-07-26　返修日期:2021-12-06
基金项目:山西省重点研发计划(201903D121121)
This work was supported by the Provincial Key Research and Development Program of Shanxi(201903D121121).
通信作者:武淑红(wushuhong@tyut.edu.cn)

Kaksonen 等[4]提出的 PROTOS 测试套件使用协议规范来产生更结构化的测试数据;Banks 等[5]针对有状态协议(如 SIP,TCP/IP 等)提出了一种名为 SNOOZE 的模糊测试工具;Devarajan[6]针对 Modbus,DNP3 等工控协议,发布了基于 Sully 工具的模糊测试模块;Vojiatzis 等[7]设计了一种 Modbus-TCP 协议模糊测试工具 MTF。现如今,一些研究已经将基于深度学习的神经网络模型融入到工控协议模糊测试中。Hu 等[8]用生成式对抗网络(Generative Adversarial Networks,GANs)训练一个基于真实协议消息的生成模型;Li 等[9]提出了基于深度对抗学习的模糊测试用例生成方法;Zhao 等[10]将长短期记忆网络(Long Short-Term Memory,LSTM)作为序列对序列(Sequence to Sequence,seq2seq)模型的编码器和解码器,以学习真实协议序列的语法并生成真实但虚假的协议消息。然而,这些先进的基于深度学习的协议模糊测试方法应用于自动化任务中时存在两个问题:1)该模型不能准确地再现细节;2)生成的协议序列消息更倾向于重复原有序列。

受此启发,本文提出了一种基于指针生成网络的模糊测试方法,该方法使用深度学习技术自动从真实协议消息序列中发现协议语法格式;并且基于该方法设计了一个自动智能的模糊测试框架,名为 PGNFuzz,该框架涵盖了工控协议消息的时间步长和特征向量,并且可以在短时间内生成大量测试协议消息。此外,PGNFuzz 不依赖于协议规范,因此可以应用于公共协议和私有协议,性能优于许多已有的框架。本文的主要贡献如下:

(1)提出了一种基于指针生成网络的方法来处理模糊测试数据的生成,该方法可以智能地自行学习协议格式规范并生成协议测试用例,并且使用覆盖机制[11]来跟踪已生成的内容,防止重复。

(2)在此基础上,建立了一种通用的工业控制网络协议模糊测试框架 PGNFuzz,该框架可以处理大多数工控协议模糊测试。

(3)为了评价本文方法的有效性,提出了一系列性能指标。实验结果表明,本文方法比 GPF 和其他基于深度学习的模糊测试方法更有效。

2 相关工作

目前,深度学习以其强大的学习能力被应用到模糊测试的各个领域。Godefroid 等[12]提出的 Learn&fuzz 使用 seq2seq 模型从一些有效的输入中学习 PDF(一种复杂的输入格式)对象的语法,并使用所学的语法来生成用于测试 PDF 解析器的测试数据。他们在微软的 Edeg 浏览器中对 PDF 解析器进行了模糊处理,虽然在实验中没有取得较好的结果,但这仍是一次很好的尝试。Rajpal 等[13]使用神经网络模型从过去的模糊测试研究中进行学习,并预测输入文件中的最佳变异位置以执行模糊突变。Nichols 等[14]利用 GANs 来帮助使用新的种子文件对系统进行初始化,提高了模糊测试框架 AFL 的性能,实验结果表明,GAN 比 LSTM 更快速、更有效,并且可以发现更多的唯一代码路径。基于深度学习的工控协议模糊化算法是利用深度学习技术来解决模糊测试用例的生成问题。Fan 等[15]使用深度神经网络来学习私有网络协议的生成输入模型,并使用学习到的模型来生成新的消息用于进行模糊测试。

虽然上述工作在一定程度上促进了工控协议检测技术的发展,但这些研究成果大多是在原有模糊测试框架的基础上进行扩展的,还存在较多的问题。这些现有的网络协议模糊测试工具大多不是针对工业控制领域的,它们不仅缺乏针对性的功能设计,还缺乏对工控协议特性的考虑。工控协议有很多种,现有的模糊测试工具所支持的 IP 数量非常有限(通常只支持 Modbus,DNP3 等),无法满足大多数 IP 的测试需求。工控协议中存在大量的私有协议,如 Conitel-2020 设备所采用的协议,这些协议规范通常不公开,也不能直接获取,因此无法理解协议的消息格式和会话过程。由于传统网络协议模糊方法对协议规范有很强的依赖性,因此在不知道协议格式规范的情况下难以实现模糊测试;许多模糊测试工具包含工控协议的通用描述方法,并且有用于通用协议的测试框架。但在许多情况下,需要测试的网络协议是私有协议,只能在小规模的私有协议上进行测试工作。私有协议的格式可能不符合通用格式,在这种情况下,传统的模糊测试工具将失败;一些模糊测试工具通过封装网络协议规范来实现对协议的测试,每次测试一个新协议时,都需要对其规范进行分析,实现规范的细节成本高且容易出错。

与传统的模糊测试工作相比,在模糊测试中应用深度学习绕过了建立协议规范和协议自动机的过程,减少了工作量。此外,基于深度学习的模糊化过程不会因为对协议规范的理解而产生逻辑错误或者人为失误,可以通过调整神经网络的参数生成模型的结构,来调整测试用例与合法输入之间的"相似性",进而在改变测试用例接受率的同时提高测试效果。

3 基于指针生成网络的测试用例生成方法

3.1 方法概述

测试用例的质量是影响模糊测试效率和有效性的重要因素。本文方法旨在从给定的工业控制网络中智能地学习原始网络数据包的格式,通过学习可以得到一个生成模型,该模型可以生成与真实数据帧相似的结构良好的测试用例,在不同的训练环境中,学习的生成模型对协议语法的理解和掌握程度不同。这项工作适用于测试大多数工控协议,本文方法主要分为 3 个步骤:1)工控协议通信数据包的预处理;2)PGN 模型构建与训练;3)模糊测试与再训练。

3.2 工控协议通信数据包的预处理

目前有各种各样的方法从不同的工控协议中捕获数据包,其中最直接的方法就是运用终端捕获工具捕获在工控网络环境中生成的工控协议数据包,并将其作为训练数据。而在深度学习中训练数据对训练模型的结果有显著影响,因此,在获得原始数据后,我们采用合适的策略对训练数据进行预处理,以此提升最终模型发现漏洞的能力。整个预处理过程主要分为 3 个步骤,具体细节如下。

(1)数据帧的聚类

模糊测试的效果主要取决于测试的深度和代码的覆盖程度。从工控协议中捕获的协议消息的长度和类型是不同的,本文模型越能理解协议消息之间的差异,就越能更好地增加测试深度。为了增强模型对消息格式的学习,我们利用帧长

聚类和 K-means 聚类等聚类策略对数据进行分类。由于具有相同长度的数据帧总是倾向于共享相同的帧类型，因此数据帧会根据其长度进行聚类；而 K-means 利用欧氏距离作为相似度的基准，更倾向于对功能相同的数据帧进行聚类，在预处理过程中采用这两种聚类方法有助于建立能够阐明数据帧结构的模型。

在训练深度学习模型时，通常使用数据扩充[16]方法来防止过度拟合，本文使用这种方法来保持生成数据的多样性。在真实协议中捕获到的部分数据帧非常少，甚至可以忽略不计。为生成数据的多样化，我们有意增加这些数据帧的比例，保持数据多样性有助于增加测试深度和测试广度，也可以提高代码覆盖率。

(2)添加特殊符号

添加特殊符号可以为后续的模型训练提供更高质量的训练数据。一般情况下，捕获数据包的过程可以分为两类，如图 1 所示。首先，对于一个已知的协议栈，例如运行 Modbus-TCP 的 TCP/IP 协议栈，可以将 IP 报头作为一个分界点，截断 IP 报头(包括 IP 源地址、目标地址和其他一些传递请求的附加信息)，并保留一个保存 IP 报头的文件，以便进一步进行数据包注入攻击。其次，在数据包最前面的位置处插入 STA(开始)作为序列开始的标志，将 END(结束)作为序列结束的标志，该操作消除了无关信息对模型的影响，并提高了捕获数据的质量。然后，若协议未知，则用地址进行学习，并且将 STA 和 END 直接添加到整个捕获数据的开头和结尾。此外，使用统一字符 PA(pad) 将短序列填充到最大帧长度，它有助于统一序列长度以便标准化训练。最后，将处理后的数据存储在训练数据集中。

(3)数据特征转换

为了让模型更好地学习到捕获的协议数据包的特征，原始的数字协议消息需要被转换为恰当的类型。受到字符嵌入机制[17]的启发，我们采用字符化预处理，首先根据如下所示的数据帧字母表将 ICS 中的协议消息的 n 个字符转换成十六进制序列。在数据帧的字母表中，有如下所示的 10 位数字字符和 6 个英文字母字符。

0 1 2 3 4 5 6 7 8 9 a b c d e f

根据字母表，协议消息序列中的每个字符都被编码为 n 维的 ***one-hot*** 向量 $\boldsymbol{x}\in R^{l\times n}$，字符在字母表中的位置是 1，其余为 0。在 ***one-hot*** 向量中包含两个维度：时间步长维度和特征向量维度。因此，长度为 l 的序列被编码成矩阵 $\boldsymbol{x}\in R^{l\times n}$，并作为字符嵌入的输入。

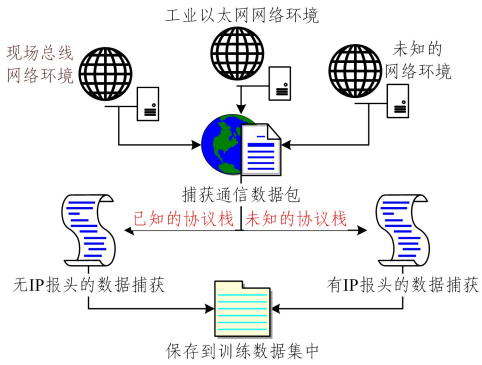


图 1 数据包捕获流程

Fig. 1 Process of capturing data packets

3.3 指针生成网络模型构建与训练

本文模型的架构中有 3 个组件，即 seq2seq 模型、注意力机制[18]和指针网络模型，架构中还引入了覆盖向量。我们的设计理念之一是在达到其效果的基础上设计轻量级模型。其凭借能减少计算资源消耗的显著特点，便于被部署到嵌入式设备上，为未来的持续在线学习奠定了基础。PGN 模型包含输入层、嵌入层、输出层、注意力机制模块和指针模块，输入和输出都是序列，不要求长度相等。此外，为减少生成序列消息的重复，还引入了覆盖机制。PGN 模型的详细结构如图 2 所示。

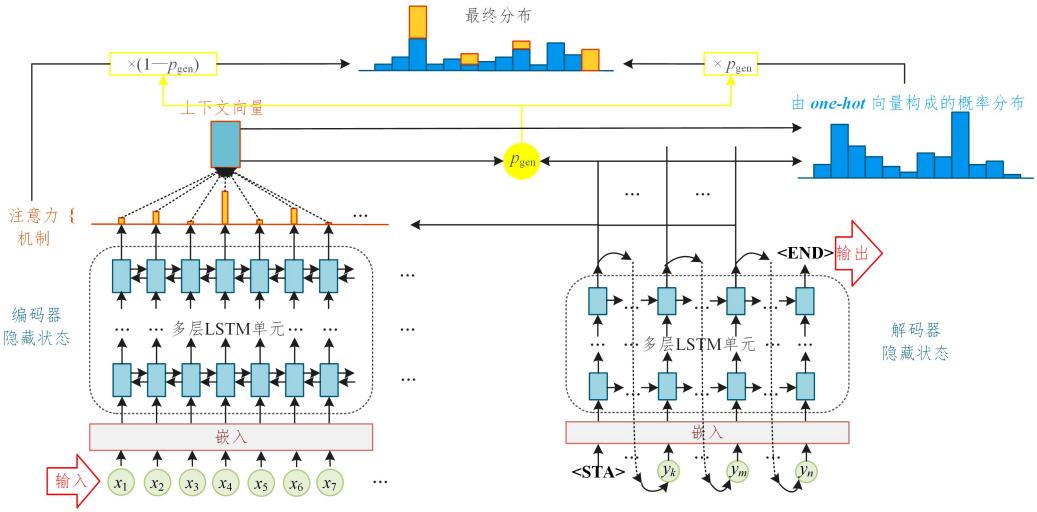


图 2 指针生成网络模型

Fig. 2 PGN model

3.3.1 模型构建

本文提出的指针生成网络模型是 seq2seq 模型和指针网络模型的结合体[19]，它既允许通过指针复制协议消息序列中的字符，也允许从给定的协议消息训练集中生成序列。在 PGN 模型中，为了整合经数据预处理后形成的矩阵上两个不同维度的特征，我们建立了具有不同层次的 LSTM 单元作为 seq2seq 模型的编码器和解码器，使得模型既能保留时间步长的维度，又能保留特征向量的维度信息。其中，编码器

采用双向 LSTM 单元学习协议消息中的字符概率分布,解码器采用单向 LSTM 单元预测学习后的协议序列,并生成用于模糊测试的带有半有效数据字段且序列语法结构良好的测试用例。

首先,训练集中的数据经预处理操作后作为字符嵌入的输入进入模型,将其逐个送入编码器 LSTM 模型中,产生一系列编码器隐藏状态 h_i。在每一步 t 上,解码器接收前一个字符嵌入产生解码器状态 S_t。输出状态 S_t 不仅用于下一个输入字符,还会转移到注意力机制模块。注意力分布可看作是序列上字符的概率分布,该概率分布引导解码器在何处产生下一个字符。

为了测量 LSTM 的堆叠效果,我们在编码器阶段和解码器阶段分别建立了 1~5 个隐藏层 LSTM 单元,每一层包含 128 个隐藏状态,并且 LSTM 层分别包含两个用于正向和反向上下文序列的子网络。由于我们的训练数据缺乏标签,因此采取无监督模式。我们观察了不同时期的模型,这些模型的执行次数代表了学习结果的优劣,以确定无监督模式下训练模型的效果。

其次,注意力分布被用于产生编码器隐藏状态的加权和,也被称为上下文向量 $\boldsymbol{h}_t^*$。

$$\boldsymbol{h}_t^* = \sum_i a_i^t \boldsymbol{h}_i \tag{1}$$

上下文向量可以被看作是在这个步骤中从训练集中读取内容的固定大小表示,它与通过两个线性层提供的解码器状态 S_t 相连接,从而产生 ***one-hot*** 向量构成的词汇表分布 P_{vocab}。

$$P_{\text{vocab}} = softmax(V'(V[s_t, h_t^*]+b)+b') \tag{2}$$

其中,V,V',b 和 b' 均为可学习参数;P_{vocab} 是所有 ***one-hot*** 向量的概率分布,为我们提供预测协议序列中字符 $\boldsymbol{w}$ 的最终分布。

$$P(w) = P_{\text{vocab}}(w) \tag{3}$$

在训练期间,时间步 t 的损失是该时间步长下目标字符 $\boldsymbol{w}_t^*$ 的负对数似然值。

$$loss_t = -\log P(\boldsymbol{w}_t^*) \tag{4}$$

则整个序列的总体损失为:

$$loss = \frac{1}{T}\sum_{t=0}^{T} loss_t \tag{5}$$

此外,根据上下文向量 $\boldsymbol{h}_t^*$、解码器状态 S_t 和解码器输入 x_t,计算时间步 t 的生成概率 $p_{\text{gen}} \in [0,1]$。

$$p_{\text{gen}} = \sigma(\boldsymbol{w}_{\boldsymbol{h}^*}^{\mathrm{T}} \boldsymbol{h}_t^* + \boldsymbol{w}_s^{\mathrm{T}} s_t + \boldsymbol{w}_x^{\mathrm{T}} x_t + b_{ptr}) \tag{6}$$

其中,向量 $\boldsymbol{w}_{h^*}$,$\boldsymbol{w}_s$,$\boldsymbol{w}_x$ 和标量 b_{ptr} 均为可学习参数,σ 是 sigmoid 函数。p_{gen} 作为软开关通过 P_{vocab} 从所有 ***one-hot*** 向量中抽样选择生成字符,或者根据注意力分布 $\boldsymbol{a}^t$ 从输入序列中复制字符。

生成的消息重复是 seq2seq 模型中的一个常见的问题[11],在生成多条消息时尤其明显,我们采用 Tu 等[11]的覆盖机制模型来解决此问题。在覆盖机制模型中,设之前所有的解码器时间步长的注意力分布之和为覆盖向量 $\boldsymbol{c}^t$。

$$\boldsymbol{c}^t = \sum_{t'=1}^{t-1} \boldsymbol{a}^{t'} \tag{7}$$

更直观地说,$\boldsymbol{c}^t$ 是语料库中消息的一个非规范化分布,它表示目前为止这些协议序列中字符从注意力机制中得到的覆盖程度。特别注意的是,$\boldsymbol{c}^0$ 是一个零向量,因为在第一个时间步中没有涉及到任何消息。

覆盖向量作为注意力机制的额外输入,确保了注意力机制当前选择的位置是参照之前选择的位置来进行决定的,这样会使注意力机制更容易避免重复关注相同的位置,从而避免产生重复的字符。

除此之外,还有必要定义一个覆盖损失,以便惩罚关注相同的位置。

$$\text{cov } loss_t = \sum_i \min(\boldsymbol{a}_i^t, \boldsymbol{c}_i^t) \tag{8}$$

需要注意的是,覆盖损失是有限的,我们的损失函数也更加灵活,因为生成的消息不需要统一的覆盖范围,我们只惩罚每个注意力分布和目前覆盖范围之间的重叠——防止注意力重复。最后,将覆盖损失以参数 λ 重新加权后再加入到原损失函数中,从而得到新的复合损失函数。

$$loss_t = -\log P(\boldsymbol{w}_t^*) + \lambda \sum_i \min(\boldsymbol{a}_i^t, \boldsymbol{c}_i^t) \tag{9}$$

3.3.2 模型训练策略

为了得到一个良好的训练模型,需采取适当的训练策略。根据 Dai 等[20]的方法,我们利用批量序列对 PGN 模型进行低维度的预训练,将预训练得到的权值作为编码器 LSTM 模型和解码器 LSTM 模型的初始化参数。其次,为 LSTM 单元选择了 Adam 随机梯度优化算法[21]。该算法通过计算梯度的一阶矩估计和二阶矩估计来计算不同参数的独立自适应学习速率,而不是保持单一的学习速率来更新所有的权值。这些策略有助于降低训练的不稳定性。

在最终模型的形成过程中,还应进行模型验证。在模型验证阶段,使用 N-Gram[22]作为判断生成序列与真实序列相似性的标准,生成的序列与真实序列共享的 N-Gram 越多,生成的序列就越好。训练阶段和验证阶段的参数是共享的,设计验证阶段的目的是评估模型性能并调整训练参数,一方面防止模型对训练数据的过度拟合,增强模型的鲁棒性,另一方面帮助我们确定最优超参数。

3.3.3 模糊测试与再训练

经过模型训练和验证,本文模型可以生成尽可能多的测试用例,向测试目标发送测试用例并记录任何可能的响应。在这个发送和接收过程中,设置一个程序模块,用于记录所有通信进程和异常行为的日志文件,该日志文件为实验分析提供了依据。除了程序监控之外,我们还需要人工分析整个过程,以发现潜在的异常。在模糊测试过程中,找出导致系统异常的数据帧序列并记录下来。在一轮模糊测试过后,利用导致异常的序列重新训练模型。为了再次进行模型训练,我们对记录下来的序列执行两个步骤。首先,对序列进行突变操作,包括单点突变和多点突变,突变操作指序列数据在一个或多个随机位置上的随机修改;其次,使用数据扩充策略来增加这些特定序列的数量。在这些步骤之后,使用更新后的训练数据集对模型进行再次训练,以期达到最佳效果。

4 PGN 框架体系结构

本文在提出的智能化模糊测试用例生成方法的基础上,又提出了一个通用型的智能化工控协议模糊测试框架,即 PGNFuzz。如图 3 所示,PGNFuzz 由消息捕获和分析模块(MCAM)、消息预处理模块(MPM)、模型训练和消息生成模块(MTMGM)、消息发送和接收模块(MSRM)、监测异常模块(MM)和日志记录模块(LM)组成,这些模块相互协作完成

整个模糊测试工作。运行 PGNFuzz 时，首先启动 MCAM 捕获训练数据，收集当前在工业控制网络环境中生成的工控协议数据包；然后，MPM 将训练数据经过预处理过程传递给 MTMGM，MTMGM 进行深度学习以训练模型。待模型训练结束后，MTMGM 利用学习后的模型生成测试数据，并由 MSRM 将数据发送到目标工业设备中。同时，MM 监督异常事件的发生，LM 记录自运行起所有发生的事件。下文将详细阐述各个组件的工作流程和实现细节。

(1)消息捕获与分析模块(MCAM)。该模块主要负责网络通信的采集与分析。首先，对工业网络环境中的流量进行实时捕获，收集通信数据包；其次，MCAM 从数据中提取设备地址并将其存储为 IP 地址和端口对，这些 IP 地址和端口将被 MSRM 在测试阶段使用；最后，MCAM 提取消息的 TCP/IP 头，提取特定于协议的命令，并以每行一条消息的格式存储为文本文件。除了包捕获之外，另一个选项是从本地文件读取数据，包括 pcap 文件和文本文件。对于 pcap 文件，MCAM 执行与捕获相同的分析步骤；对于文本文件，PGNFuzz 不打算对它们做任何事情，而是直接发送给 MPM。

(2)消息预处理模块(MPM)。该模块的任务是提供良好的训练数据。MPM 首先将异常数据从原始数据集中移除，再用上述的数据包预处理方法对捕获到的协议消息进行预处理。MPM 将预处理后的训练数据发送到 MTMGM。

(3)模型训练和消息生成模块(MTMGM)。该模块是核心模块，其工作任务是深度学习模型训练和测试用例生成。该模块实现了本文提出的测试用例生成方法。MTMGM 使用反向传播算法对 PGN 模型进行训练，采用 Adam 随机梯度优化算法对参数进行更新。在大多数情况下，MTMGM 都可以直接使用默认配置，但为了获得更好的学习效果，建议设计具体的模型结构，使用有利的训练设置。

MTMGM 的训练过程如下：MTMGM 首先以数值表示的形式加载所有训练数据，然后根据训练设置执行训练。由于训练总是需要很长时间，并且在此期间可能会发生异常，因此 MTMGM 会自动地将模型参数导出到检查点文件中，以防止前面的工作被浪费。如果在训练期间发生任何异常，PGNFuzz 将从最近的系统检查点恢复。另外，MTMGM 根据设置也会将模型参数转储到检查点文件。

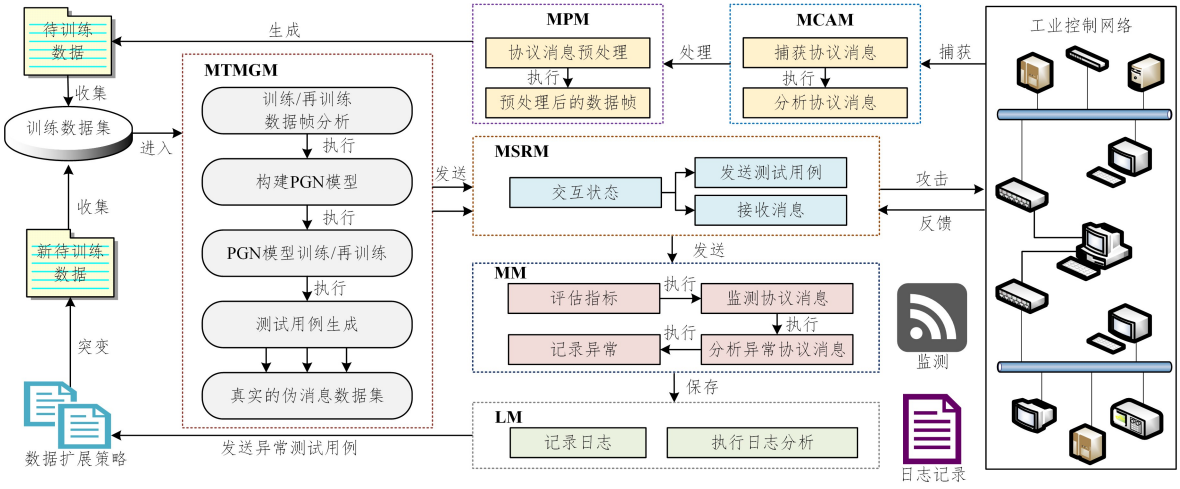


图 3 PGNFuzz 框架结构

Fig. 3 PGNFuzz framework

训练之后，MTMGM 通过从不同的检查点文件读取参数，来使用不同的模型生成测试用例。所有生成的测试用例将存储在文本文件中，然后由 MSRM 块读取。

(4)消息发送和接收模块(MSRM)。该模块主要负责向工业控制网络发送报文并接收反馈消息。一方面，MTMGM 生成的每个测试用例都被交付给了 MSRM。MSRM 使用 MCAM 获得的地址信息，用 TCP/IP 头包装这些数据发送给目标。另一方面，MSRM 接收工业设备的响应。当生成的测试用例发送完成后，所有通信将被传送到 LM。

(5)监测异常模块(MM)。在模糊测试工作过程中监控工控网络的异常，主要是通过监测和分析交互数据来实现的。MM 试图找出发送报文后是否有任何故障和异常发送。此外，为了获得更好的模糊测试结果，人工监测也是必要的，因为一些异常可能在系统运行过程中没有直接的提示，只能通过观察其运行来监测。在通信的过程中，通过 MM 和 MSRM 的协作，将正常通信数据和发生的异常记录发送给 LM。MM 将以下性能指标用于异常消息监测分析。

1)测试用例识别率(TCRR)

TCRR 指测试目标所能识别的测试用例的百分比。更具体地说，即为发送给测试目标的有效测试用例的比例。TCRR 越高表明生成的测试用例在格式上与真实协议的相似度越高，说明模糊测试的深度得到了保证。相反，TCRR 越低则意味着测试目标会丢弃更多的测试用例，表明模型需要调整或重新训练。其计算式如下：

$$TCRR=\frac{nRecognized}{nSent}\times 100\% \tag{10}$$

其中，$nRecognized$ 指测试目标所能识别的测试用例的数量，$nSent$ 指发送给测试目标的测试用例总数。

2)触发异常效率(TAE)

TAE 反映了发送固定数量的测试用例后触发异常的次数和时间，其值越大表示触发异常的能力越强。由于模糊测试的最终目标是在短时间内发现尽可能多的漏洞，因此不仅要考虑测试过程中是否能发现异常，还要考虑测试效率。应注意的是，发现异常的数量与测试目标也有关。然而，本研究

只关注方法的效率。其计算式如下：

$$TAE=\frac{nAnomalies}{nAllCases}\times 100\%+\frac{\partial}{\sum_{k=1}^{M} t_i} \tag{11}$$

其中，$nAnomalies$ 指发现异常的数量，$nAllCases$ 指所有测试用例的数量，$t_{\text{anomalies}}$ 记录上一次正常请求发起到下一次异常反馈的间隔（丢弃 5 个最大值和 5 个最小值），M 是时间间隔总数，t_i 是发现 i_{th} 异常的时间间隔，$t_{\text{anomalies}}=[t_1,t_2,\cdots,t_m]$，$\partial$ 是时间间隔总和的倒数的权重。

3）生成数据的多样性（DGD）

DGD 指生成训练数据多样性的能力，生成的数据帧越多样越容易引起更多的异常，这也就意味着代码覆盖率越高。具体表达式如下：

$$DGD=\frac{nGCategory}{nACategory}\times 100\% \tag{12}$$

其中，$nGcategory$ 指生成数据帧中的消息类型的总数，$nACategory$ 指训练集中消息类型的总数。

（6）日志记录模块（LM）。该模块记录了测试工作过程中发生的所有事件。MSRM 一旦向工业网络中发出第一个测试用例，LM 就会被激活，然后 LM 按顺序记录每个测试用例及其返回值。同时，LM 也会接收 MM 的信息，MM 监测到的任何异常都会被记录到日志文件中。此外，LM 还会专门挑出触发了系统异常或不当行为的消息，以供下一次模型训练使用。

5 实验评估

5.1 训练数据

（1）Modbus-TCP

为了从不同的维度分析实验结果，在实验中使用了按照 Modbus 协议规范设计的不同的调试工具，包括 Modbus RSSim v8.20，Modbus Slave v6.0.2 和 xMasterSlave v.156 等，用于帮助我们进行 Modbus 协议模糊测试。此外，为了更好地证明本文方法的有效性，我们在 MCU[23] 和 PC 之间部署了串行通信模式，并采用 RS485 总线[24] 进行信号传输，构建了真正的 Modbus-TCP 网络环境。生成的测试用例被发送到真实环境中，以测试实际应用中的效果。

我们使用一个可以实现 Modbus 协议的 python 包——Pymodbus 来生成训练数据帧，通过其调用相应的函数，以快速便捷地生成足够多的不同类型的数据帧。我们随机调用程序中的函数来生成各种类型的数据，共计 30 万条，并将其放入 PGNFuzz 的 MCAM 模块的目录中。

（2）EtherCAT

为了捕获 EtherCAT 通信数据包，我们模拟了一种基于 ICS 的 EtherCAT 网络环境，如图 4 所示。主站为 Beckhoff 工业 PC，从站包括 Beckhoff 总线耦合器 EK1100、数字输入 EL1004 和数字输出 EL2004。并且使用 ET2000 作为主机和从机之间的在线监听器。Wireshark 在终端设备中从监听器获取并显示大量的通信数据消息。经过 MPM 处理后，这些消息被作为 PGNFuzz 的训练数据。此外，这些工具对实际工业网络环境的影响很小，是捕获网络协议消息的理想选择。

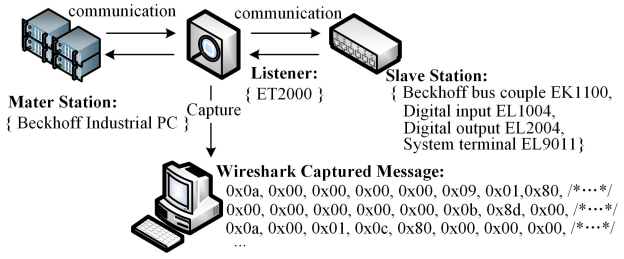


图 4 EtherCAT 环境

Fig. 4 EtherCAT environment

5.2 实验设置

我们采用当下流行的深度学习框架 TensorFlow 来实现模型架构。为了提高训练效率，实验在配置为八核处理器（Intel(R) Core(TM) i7-6700K CPU@4.00 GHz），16 GB RAM，Nvidia GeForce GTX 1080 Ti(11GB)的 64 位操作系统为 Windows 的机器上训练模型。

将批量输入的特征序列作为设计模型的输入进行预训练，并获得正式训练的初始化参数。输入的特征数据分为 3 部分：训练数据、验证数据和测试数据。训练和验证部分完成后，得到可以生成半有效序列的相关参数。我们利用 Adam 算法来训练多个周期（周期为 1,3,8,13,18,23,28）的特征数据，并为不同周期的训练保存数据。其中，学习速率为 0.15，初始累加值为 0.1。除此之外，还设置了阈值为 2 的梯度裁剪，但不使用任何形式的正则化。为了获得最终的覆盖模型，我们添加了损失加权为 $\lambda=1$ 的覆盖机制（见式（9）），并进行 3 000 次迭代（约 2 h）以进行进一步的训练。在这段时间内，覆盖损失从最初的 0.5 收敛到 0.2 左右。我们还尝试了 $\lambda=2$ 时的情况，这虽然减少了覆盖损失，但增加了主要损失函数，因此本文并未使用它。

5.3 实验结果

本小节分为 3 个部分来说明实验结果。首先给出并分析 Modbus-TCP 模糊测试的结果，评估所实现的模型的有效性和效率。在此基础之上，分析了在模糊测试过程中出现的一些特殊异常。最后给出了 EtherCAT 协议的实验结果，证明了框架的通用性。

5.3.1 统计分析及结果

本文选择广泛使用的 GPF 以及基于 GAN 模型、基于 LSTM 的 seq2seq 模型模糊器进行对比实验。测试的目标是上述的 3 种 Modbus 仿真软件以及我们构建的真实 Modbus 网络环境。为了更好地评估模型对协议的总体效果，我们结合了 4 个系统的实验结果。在所有的数据中每个模糊器得到的数据权重均为 25%，在所有模型经过充分训练之后，通过 TCP/502 端口将生成的 270 000 个测试用例发送到 Modbus 中进行模糊测试。

利用上述的 3 个评价指标对 PGNFuzz 框架和其他 3 种模糊器的有效性和效率情况进行了评估，并以图形化的方式进行表示，具体情况如下所述。

（1）TCRR

TCRR 的实验结果如图 5 所示。由于 GPF 不涉及连续的学习过程，因此与其他两种基于深度学习的模糊器相比，GPF 的性能对目标没有变化趋势，用一条水平线表示。从这

5个模型来看，TCRR指标的性能水平为：GPF≈基于LSTM模型＜基于GAN模型＜基于PGN模型。

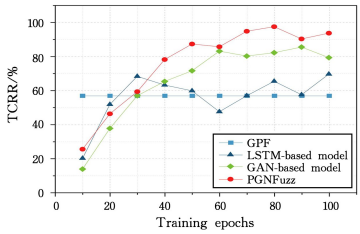


图5 TCRR随着训练次数变化而变化

Fig. 5 TCRR changes with training epochs

经过30多次的训练，基于GAN模型的模糊器的TCRR率明显超过了GPF，在训练60次之后，其TCRR率增长趋势明显放缓。GPF的平均TCRR率为57%，而基于GAN模型的模糊器最终的TCRR率为75%～90%。基于LSTM的seq2seq模型的模糊器最终的TCRR率明显低于基于GAN模型的模糊器，这可能是因为无法有效学习数据表示的层次结构所导致的。由于基于LSTM的seq2seq模型是在无监督学习环境中进行训练，因此可能会导致生成大量格式错误的协议消息序列，从而导致生成异常协议消息的比例迅速上升，这也是造成TCRR率大幅波动的主要原因。协议报文消息的功能码和参数范围有限，因此增加了随机生成消息时异常识别的可能性。

我们认为，基于LSTM的seq2seq模型是一个很好的方法，可视为本文的基线模型，在其模型基础上引入注意力机制和指针网络，可以突破原模型的瓶颈。基线模型使用softmax函数来预测生成，我们则通过计算生成概率 p_{gen} 从语料库中抽样选择生成字符，这有助于一次性提高或降低所有生成字符或者所有复制字符的概率，而不是单独提高或降低某一项的概率。其次，利用注意力分布引导解码器在何处产生下一个字符，通过设置注意力机制可以将模型有限的注意力集中在重点信息上，从而节省资源，更加快速地获得最有效的消息，而基线模型并未设置注意力机制。我们注意到，指针网络经常复制一个字符，同时该字符在语料库中多次出现，通过计算之前所有的解码器时间步长的注意力分布之和来获得覆盖向量，以避免注意力机制重复关注相同的位置，从而避免生成消息重复的问题，基线模型则未考虑此问题。

基于GAN模型的模糊器只考虑利用文本的时间步长来获得定长向量，忽略了空间结构特征。然而，时间步长维度和特征向量维度并不是相互独立的，将协议消息序列中的每个字符都编码为 n 维的 ***one-hot*** 向量，通过字符嵌入层将其转换为模型的输入，这样不仅能保留时间步长维度，还能保留特征向量维度。本文模型可以较为准确地再现协议规范细节，大幅度提升性能。当训练次数为50～100时，PGNFuzz的平均TCRR率比基于GAN模型的模糊器高出约8%，而比基于LSTM的seq2seq模型的模糊器高出约13%，间接说明了PGNFuzz更适用于工控协议测试用例的生成。

(2)ATE

在用Modbus协议进行模糊测试时，我们记录利用模型进行模糊测试触发异常的类别(CTA)、数量(NTA)、平均时间间隔(ATITA)、真负率(TNR)和真正率(TPR)等数据。我们将4种模型与以上统计的数据或计算得出的数据进行比较，以便更好地说明和评估ATE指标。为此，我们在4个模糊测试目标上测试了4种模型，发送了由每个模型产生的27000条假数据。PGNFuzz与其他3种模型的测试结果如表1所列。

表1 实验数据比较

Table 1 Experimental data comparsion

Test Model	Case Amount	Training Time/h	Test Time/h	Targets	CTA	NTA	ATITA	TNR	TPR	ATE Score
PGNFuzz	270000	15.54	12.31	Modbus RSSim v8.20	6	198	0.72	93.86	90.81	1.31
				Modbus Slave v6.0.2	4	57	2.01			0.99
				xMasterSlave v.156	7	101	1.88			1.23
				Real Environment	6	42	3.23			1.07
GAN-based model	270000	8.57	9.52	Modbus RSSim v8.20	5	86	1.66	92.54	92.39	1.09
				Modbus Slave v6.0.2	6	57	2.50			0.81
				xMasterSlave v.156	5	62	2.30			0.94
				Real Environment	4	23	6.21			0.72
LSTM-based model	270000	5.18	9.51	Modbus RSSim v8.20	4	38	3.76	87.39	93.25	0.76
				Modbus Slave v6.0.2	5	21	6.78			0.63
				xMasterSlave v.156	4	18	7.93			0.52
				Real Environment	3	14	10.20			0.47
GPF	270000	29.05	9.47	Modbus RSSim v8.20	2	23	6.21	—	—	0.16
				Modbus Slave v6.0.2	2	8	17.84			0.12
				xMasterSlave v.156	3	12	11.89			0.19
				Real Environment	2	9	15.86			0.08

PGNFuzz在3种Modbus调试工具以及真实环境中进行Modbus协议模糊测试，都能得出比其他模型更好的结果。以往的一些模糊测试技术只注重生成的协议消息能否触发异常，忽视了触发异常的效率。本文模型在对目标进行模糊测试时，不仅考虑了在测试中能否发现异常，还考虑了模糊测试的效率。从CTA列的比较表明，对于测试用例的多样性而言，PGNFuzz模型在处理相同协议时比其他3种模型的性能更好。PGNFuzz是基于LSTM模型的扩展，从本文方法在测试集上取得的结果可知，PGNFuzz可以捕获更多协议消息中的依赖关系。实验结果表明，本文模型在4个模糊测试目标上的ATE得分均为最高。在4个测试目标中，PGNFuzz在Modbus RSSim v8.20上的表现是最好的，这意味着可能

这是一个较弱的测试目标。

其中，PGNFuzz 的具体表现如表 2 所列，表 2 列出了 PGNFuzz 针对 4 个模糊测试目标的具体实验数据。其中，NTP 表示触发测试目标的数量，其也是模糊测试代码覆盖率的度量标准。根据 Modbus/TCP 协议数据字段的特点，由表 2 的第一列可以得到本文方法触发的协议异常特征描述。从表 2 可以看出，与表 1 中其他深度学习方法的结果相比，本文的模糊测试方法不仅可以保证发现漏洞的能力，而且增加了发现漏洞的频率，测试效率更好。

表 2 PGNFuzz 触发异常情况

Table 2 Triggered anomalies of PGNFuzz

Triggered Anomalies	Frequency (Times)	ATITA (Mins)	NTP
Slave crash	23	24.25	4
Station ID xx off-line	128	3.86	3
Using abnormal function code	96	4.13	4
Automatically closes the window	36	14.57	4
Data length unmatched	101	3.89	4
Abnormal address	37	23.26	3
Integer overflow	5	108.00	2
File not found	3	197.00	1

(3) DGD

GPF 学习协议消息的类别是固定的，不同的 GPF 的覆盖范围是不同的，因此本文不进行讨论。原始数据集中共有 11 种不同类别的 Modbus 数据帧，当训练次数为 10 时，3 种基于深度学习的模型的 DGD 保持得最好。再经过多次训练，通常会丢失一些消息类别，如图 6 所示。

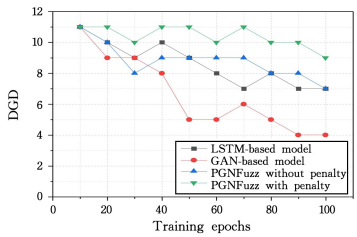


图 6 DGD 随着训练次数变化而变化

Fig. 6 DGD changes with training epochs

基于 LSTM 模型和 PGNFuzz 在保持生成测试用例多样性方面表现良好，说明这两种模型能够学习协议消息的时间步长。PGNFuzz 由于在基于 LSTM 模型的基础上增加了注意力机制、指针网络和覆盖机制，因此可以更好地利用过去和未来的消息，其性能比基于 LSTM 模型更好。通常数据类型越丰富，我们可能实现的代码覆盖率就会越高，模型检测出异常的能力也就越强。由此可见，如表 2 所列，本文模型可以有效检测出各种异常。

5.3.2 异常分析

第一个具有代表性的异常是当 Modbus RSSim 被生成的数据帧攻击时，仿真软件控制台会显示出大量的异常信息。当我们发送大约 3500 个数据帧后，软件弹出提示框表示程序崩溃了。我们又将生成的 3450～3500 个数据帧发送到 Modbus Slave 和 xMasterSlave 中，这两个 Modbus 仿真软件没有发现异常。这种比较表明，Modbus RSSim 在模拟 Modbus-TCP 协议中存在一些错误，在测试 Modbus RSSim 时偶尔会出现“文件未找到”的错误，这是由于程序没有正确处理文件操作而导致的，但这显示了本文方法揭示软件错误的能力。

另一个值得讨论的异常是，对 Modbus Slave 发送大约 6540 个数据帧后，日志中显示“Station ID 36 off-line，no response sent”。但是我们观察到，Station ID 36 仍然在线，因此认为从属状态判断可能存在执行缺陷。为了验证我们的猜测，我们向其余两个 Modbus 仿真软件发送了同样的命令，但并没有发现异常。在此之后，我们随机修改导致异常的消息单元字段，其他位置保持不变，然后将其发送给 Modbus Slave。此时，Modbus Slave 可以正确地处理这些数据帧。为了排除在测试期间“Station ID XX”离线的可能性，我们重新启动 Modbus Slave 并向其发送相同的消息，并得到相同的返回。这些分析证明了我们的猜测是正确的。

第三个特殊的异常是，在对 xMasterSlave 进行模糊测试时发现，该软件有时会自动关闭窗口。通过对系统日志的分析发现，内存溢出是导致软件崩溃的原因，这表明程序员在实现仿真器时可能未考虑到数据边界填充的情况。

在对 3 个仿真软件和真实环境进行进一步的模糊测试攻击中，除了上述提到的异常情况，我们还发现了“使用异常函数代码”“数据长度不匹配”“整数溢出”和“地址异常”等异常情况。由于这些异常具有普遍性且在之前的研究中也进行了说明，因此只记录了触发这些异常的测试用例，并统计了来自 3 个软件和真实 Modbus TCP 网络环境的所有触发异常情况，以供对本文模型进行再次训练。需要说明的是，不同的异常行为可能是由相同的原因引起的。在我们的实验中，由于测试目标没有源代码，因此未进一步区分每个异常背后的根本原因。

5.3.3 PGNFuzz 在 EtherCAT 中的应用

表 3 列出了通过重新训练的 PGNFuzz 在 EtherCAT 协议中检测到的潜在的漏洞，包括 MITM 攻击、MAC 地址欺骗、从地址攻击、数据包注入等。

表 3 在 EtherCAT 中出现的潜在漏洞

Table 3 Potential vulnerabilities in EtherCAT

Potential Vulnerabilities	NTA	Sent Number
Packet injection attack	118	30000
Man in the middle attack	26	30000
Working counter attack	209	30000
MAC address spoofing	41	30000
Slave address attack	13	30000
Unknown attack	197	30000

在实验中，我们将生成的协议消息 S_i 发送到从站并记录相应的接收到的消息 R_i，从而获取到大量的消息对$\langle S_i, R_i \rangle$。根据上面的异常协议特征，我们对指定字段值进行分析比较，得到如下的统计结果。在 EtherCAT 协议上的实验表明，该方法有很强的通用性。

结束语 本文提出了一种有效的模糊测试方法和模糊测试框架，该方法在不知道详细协议规范的情况下，利用指针生成网络来学习真实工控协议消息序列并生成真实但虚假的数据帧。最后，对 Modbus-TCP 和 EtherCAT 两种安全且运用广泛的工控协议进行了模糊测试评价。结果表明，本文方法具有良好的可扩展性，可以应用于一系列的工控协议。

在未来的研究中，我们希望构建一个可以部署在嵌入式设备中的更智能且更自动化的网络协议模糊测试系统。该系统是轻量级的，可以采用在线学习的方式，自动学习不同协议

的规范和消息格式。考虑到目前的情况，可以进一步探索其他架构以增强本文方法，如 WGAN-GP，Transformer-XL 等。其次，需要更加智能的日志分析方法和结果分析策略来提高统计分析的效率。我们最终的研究目标是整合每个优秀的体系结构和处理模块，形成一个混合模型和一个完整的软件系统，为 PGNFuzz 开发一个良好的用户交互界面，使其更易于使用。

参 考 文 献

[1] KIM S K, KOPPEN M, BASHIR A K, et al. Advanced ICT and IOT Technologies for the fourth Industrial Revolution [J]. Intelligent Automation & Soft Computing, 2020, 26(1): 83-85.

[2] WAN M, LI J, LIU Y, et al. Characteristic insights on industrial cyber security and popular defense mechanisms [J]. China Communications, 2021, 18(1): 130-150.

[3] MILLER B P, FREDRIKSEN L, SO B, et al. An empirical study of the reliability of UNIX utilities[J]. Communications of the ACM, 1990, 33(12): 32-44.

[4] RAULI K, MARKO L, ARI T. Software security assessment through specification mutations and fault injection [C]// Communications and Multimedia Security Issues of the New Century. New York: Springer, 2001: 173-183.

[5] GREG B, MARCO C, VIKTORIA F. Snooze: toward a stateful network protocol fuzzer [C]// International Conference on Information Security. New York: ACM, 2006: 343-358.

[6] DEVARAJAN G. Unraveling SCADA protocols: using sulley fuzzer, presented at the DefCon 15 Hacking conference [EB/OL]. http://www.defcon.org/html/defcon-15/de-15-speakers.html.

[7] VOVIATZIS A G, KATSIGIANNIS K, KOUBIAS S. A Modbus/TCP Fuzzer for testing internetworked industrial systems [C]// 2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA). IEEE, 2015.

[8] HU Z C, SHI J Q, HUANG Y H, et al. GANFuzz: A Gan-based industrial network protocol fuzzing framework [C]// The 15th ACM International Conference, Computing. Frontiers. New York: ACM, 2018: 138-145.

[9] LI Z H, ZHAO H, SHI J Q, et al. An Intelligent Fuzzing Data Generation Method Based on Deep Adversarial Learning [J]. IEEE Access, 2019, 7: 49327-49340.

[10] ZHAO H, LI Z H, WEI H S, et al. SeqFuzzer: An Industrial Protocol Fuzzing Framework from a Deep Learning Perspective [C]// 2019 12th IEEE Conference on Software Testing, Validation and Verification. Xi'an, China: ICST, 2019: 59-67.

[11] TU Z P, LU Z D, LIU Y, et al. Modeling coverage for neural machine translation [C]// Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. Berlin: ACL, 2016: 76-85.

[12] GODEFROID P, PELEG H, SINGH R. Learn&fuzz: Machine learning for input fuzzing [C]// Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering. Urbana: IEEE Press, 2017: 50-59.

[13] RAJPAL M, BLUM W, SINGH R. Not all bytes are equal: Neural byte sieve for fuzzing [EB/OL]. https://arxiv.org/abs/1711.04596.

[14] NICHOLS N, RAUGAS M, JASPER R, et al. Faster fuzzing: Reinitialization with deep neural models [EB/OL]. https://arxiv.org/abs/1711.02807.

[15] FAN R, CHANG Y. Machine learning for black- box fuzzing of network protocols [C]// International Conference on Information and Communications Security. Beijing: ICICS, 2017: 621-632.

[16] JUSTIN S, JUAN P B. Deep convolutional neural networks and data augmentation for environmental sound classification [J]. IEEE Signal Process Letters, 2017, 44(3): 279-283.

[17] LEVY O, GOLDBERG Y. Neural word embedding as implicit matrix factorization [C]// Proceedings of the 28th International Conference on Neural Information Processing Systems. Montreal: NIPS, 2014: 2177-2185.

[18] DZMITRY B, KYUNGHYUN C, YOSHUA B. Neural Machine Translation by Jointly Learning to Align and Translate [C]// 3rd International Conference on Learning Representations. San Diego: ICLR, 2017.

[19] ORIOL V, MEIRE F, NAVDEEP J. Pointer networks [C]// Proceedings of the 29th International Conference on Neural Information Processing Systems. Montreal: NIPS, 2015.

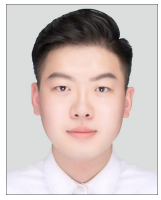
[20] DAI A M, LE Q V. Semi-supervised sequence learning [C]// Proceedings of the 28th International Conference on Neural Information Processing Systems. Montreal: NIPS, 2014: 3079-3087.

[21] KINGMA D, BA J. ADAM: A method for stochastic optimization [C]// the 3rd International Conference for Learning Representations. San Diego: ICLR, 2015.

[22] DODDINGTON G. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics [C]// Proceedings of the Second International Conference on Human Language Technology Research. San Francisco: HLT, 2002: 138-145.

[23] ROBERTS JR JD, IHNAT J, SMITH JR W. Microprogrammed control unit (MCU) programming reference manual [C]// ACM Sigmicro Newsletter. 1972: 18-57.

[24] FENG Z L, YU J X. Design and implementation of rs485 bus communication protocol [J]. Computer Engineering, 2012, 38(20): 215-218.



WANG Tian-yuan, born in 1997, postgraduate, is a student member of China Computer Federation. His main research interests include vulnerability mining, information security and machine learning.

WU Shu-hong, born in 1969, Ph.D, associate professor, master supervisor. Her main research interests include embedded systems, intelligent information processing, brain informatics and information security.

（责任编辑：喻藜）