计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200800030

IoTGuardEye:一种面向物联网服务的 Web 攻击检测方法

刘 新 黄缘缘 刘子昂 周 睿
兰州大学信息科学与工程学院 兰州 730000
(xliu2019@lzu.edu.cn)

摘 要 在包括物联网(Internet of Things,IoT)设备的绝大部分边缘计算应用中,基于互联网应用技术(通常被称为 Web 技术)开发的应用程序接口(Application Programming Interface,API)是设备与远程服务器进行信息交互的核心。相比传统的 Web 应用,大部分用户无法直接接触到边缘设备使用的 API,使得其遭受的攻击相对较少。但随着物联网设备的普及,针对 API 的攻击逐渐成为热点。因此,文中提出了一种面向物联网服务的 Web 攻击向量检测方法,用于对物联网服务收到的 Web 流量进行检测,并挖掘出其中的恶意流量,从而为安全运营中心(Security Operation Center,SOC)提供安全情报。该方法在对超文本传输协议(Hypertext Transfer Protocol,HTTP)请求的文本序列进行特征抽取的基础上,针对 API 请求的报文格式相对固定的特点,结合双向长短期记忆网络(Bidirectional Long Short-Term Memory,BLSTM)实现对 Web 流量的攻击向量检测。实验结果表明,相比基于规则的 Web 应用防火墙(Web Application Firewall,WAF)和传统的机器学习方法,所提方法针对面向物联网服务 API 的攻击具有更好的识别能力。

关键词:威胁感知;双向长短期记忆;边缘计算;Web 攻击;物联网
中图法分类号 TP393

IoTGuardEye:A Web Attack Detection Method for IoT Services

LIU Xin,HUANG Yuan-yuan,LIU Zi-ang and ZHOU Rui
School of Information Science& Engineering,Lanzhou University,Lanzhou 730000,China

Abstract In most of the edge computing applications including Internet of Things (IoT) devices,the application programming interface (API) based on Internet application technologies,which are commonly known as Web Technologies,is the core of information interaction between devices and remote servers. Compared with traditional web applications,most users cannot directly access APIs used by edge devices,which makes them suffer fewer attacks. However,with the popularity of edge computing,the attack based on API has gradually become a hot spot. Therefore,this paper proposes a web attack vector detection method for IoT service providers. It can be utilized to detect malicious traffic against its API services and provide security intelligence for the security operation center (SOC). Based on the feature extraction of text sequence requested by hypertext transfer protocol (HTTP),this method combines bidirectional long short-term memory (BLSTM) to detect the attack vector of web traffic according to the relatively fixed format of API request message. Experimental results show that,compared with the rule-based Web application firewall (WAF) and traditional machine learning methods,the proposed method has better recognition ability for attacks on IoT service APIs.

Keywords Threat awareness,BLSTM,Edge computing,Web attack,Internet of Things

1 引言

随着边缘计算技术与 5G 等基础通信技术的快速普及,在内容分发网络(Content Delivery Network,CDN)等传统边缘计算需求显著增加的同时,以物联网设备为代表的边缘计算设备数量呈爆炸式增长。据 Gartner 估算,2020 年接入互联网的设备会达到 250 亿以上[1],其中大部分都是物联网设备。与此同时,赛门铁克公司发布的《互联网安全报告》显示,2018 年针对服务终端设备的 Web 攻击增长了 56%,日均拦截数超过 130 万次[2]。

基于 Web 实现的 Restful API 是目前被物联网设备、个人计算设备广泛使用的数据交互基础技术,然而现有的 Web

到稿日期:2020-08-05 返修日期:2020-11-25 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家重点研发计划资助(2020YFC0832500);国家自然科学基金项目(61402210);教育部-中国移动科研基金项目(MCM20170206);国家电网公司科技项目资助(SGGSKY00WYJS2000062)
This work was supported by the National Key R&D Program of China(2020YFC0832500),National Natural Science Foundation of China (61402210),Ministry of Education-China Mobile Research Foundation(MCM20170206) and State Grid Corporation of China Science and Technology Project(SGGSKY00WYJS2000062).
通信作者:周睿(zr@lzu.edu.cn)

安全设计大多针对传统的 Web 服务实现，缺乏对 API 的针对性优化，加之物联网行业正处于蓝海状态，设备厂商的安全技术水平良莠不齐，这都使得物联网服务的数据面临严重的安全威胁。

本文提出了 IoTGuardEye 这一针对物联网服务的 Web 攻击检测方法，该方法在对 HTTP 请求的文本序列进行特征抽取的基础上，针对 API 请求报文格式相对固定的特点，基于 BLSTM 网络为不同的 API 训练不同的识别模型，实现了对 Web 流量的高性能攻击的向量检测。本文的主要贡献如下：

(1)针对物联网服务中 API 格式相对固定、变化相对较小的特点，设计了专用的识别模型，在无需复杂规则设计和人工介入的条件下，该模型达到了良好的攻击检测效果，具备发现零日漏洞攻击的能力，为安全运维中心提供了安全情报支撑。

(2)设计了 Web 流量特征提取、基于 API 的自动模型选择和模型训练方法，并基于此设计实现了分布式的系统架构方案。

(3)在真实的攻击流量数据集上进行实验，结果证明本文基于 BLSTM 设计的检测方法相比已有方案具有更好的检测效果。

2　相关工作

在攻防对抗中，攻击检测技术一直是热点研究内容之一，也被称为入侵检测技术。Anderson 等于 1980 年提出了入侵检测系统的概念[3]。Denning 于 1987 年提出的专家系统被认为是最早的入侵检测系统[4]。此后，研究者又基于正则表达式方法[5]、特征签名方法[6]等实现了不同的攻击检测方案。这类方案具有较高的准确度，有效提高了攻击检测系统的能力，但由于规则的滞后性，这类方案依然没有解决攻击者利用未知的攻击载荷绕过检测系统的问题。

后续又有一些研究者尝试利用机器学习带来的推理、泛化能力来解决绕过问题。例如，Kruegel 等[7]利用决策树来优化特征签名的检测效果；Chen 等[8]尝试通过支持向量机(Support Vector Machine，SVM)来识别攻击流量。随着近年来深度学习技术的兴起，基于循环神经网络(Recurrent Neural Network，RNN)的攻击检测成为研究热点。Liang 等[9]为解决基于异常的 Web 攻击检测方法准确性不高、通用性差等问题，首次将 RNN 应用于建立基于异常的 Web 攻击检测系统。Saxe 等[10]通过分析原始数据、人工选择特征来训练深度学习模型，从而检测恶意软件。

文献[11]提出了一种基于分布式深度学习来进行统一资源定位器(Uniform Resource Locator，URL)分析的 Web 攻击检测系统，其可以被部署在边缘设备上。一些基于异常的 Web 流量入侵检测往往涉及分类器集成，然而由于整体设计不佳，其表现并不理想。针对这一现状，文献[12]提出了一种新的基于异常的 Web 攻击检测系统的分层集成方案。文献[13]提出了一种利用注意力的 seq2seq 网络来进行 Web 攻击检测的新方法。该方法通过预测 Web 请求的响应并计算其与实际响应的差异，成功地对流量进行了分类。文献[14]提出了一种基于 LSTM(Long-Short Term Memory)的深度神经网络模型 DeepLog，该模型可将系统日志建模为自然语言序列。DeepLog 能够从正常执行中自动学习日志模式，并在日志模式偏离正常模型时检测出异常。文献[15]提出了基于 LSTM-RNNs 的多信道智能攻击检测方法。由于 Web 应用通常使用 HTTP 协议，因此文献[16]提出了基于 HTTP 包负载文本的攻击检测方法，该方法结合了自动编码器(Auto-Encoder)和 RNN。Skaruz 等[17]利用 RNN 在 SQL 语句级别进行 SQL 注入攻击(SQL Injection，SQLi)识别，Liu 等[18]利用 RNN 来识别攻击载荷，这两种方法均取得了良好的效果。因此，本文选择 RNN 作为识别模型的基础网络是较为可靠的。

3　数据预处理与检测模型

3.1　数据预处理

Web 流量绝大部分是通过 HTTP/HTTPS 协议进行传输的，通常通过 URL 参数或 POST Body 将数据传输给服务端程序。如果服务端在处理数据的过程中存在漏洞，那么攻击者就可以利用这些漏洞完成攻击。本文主要研究这类攻击方法，包括 SQLi、跨站攻击脚本(Cross-Site Scripts，XSS)、本地文件包含(Local File Inclusion，LFI)和远程代码执行(Remote Code Execution，RCE)。表 1 列出了这 4 种攻击类型常见的攻击载荷。

表 1　攻击的示例

Table 1　Attack examples

Type	URL Encoded Request	Payload	Description
Normal	http://victim.com/?s=test	Test	Normal HTTP Request
SQLi	http://victim.com/?s=%27)%20and%20(%271%27%3d%271	') and ('1'='1	Boolean-based SQL Injection
LFI	http://victim.com/?ReadFile=..%2f..%2f..%2fetc%2fpasswd	../../../etc/passwd	Local Confidential File Inclusion
XSS	http://victim.com/?username=%3cscript%3ealert(1)%3c%2fscirpt%3e	¡script¿alert(1)¡/script¿	Reflected Cross-Site Script
RCE	http://victim.com/?folder=test%26%26cat%20%2fetc%2fpasswd	test&&cat /etc/passwd	Bash Command Injection

由表 1 可知，正常的请求会传输 Key-Value 对给远程服务器，且不包含任何干扰远程服务端程序的载荷。但是，在攻击的示例中可以明确地看出载荷在尝试误导服务端程序，如 SQLi 中的载荷试图让服务端的程序将其作为 SQL 语句的一部分来执行。

HTTP 协议可以通过请求参数、Body 和 Headers 这 3 种方式来传递数据。请求参数被编码并记录在统一资源标识符(Uniform Resource Identifier，URI)中。Body 则包含多种数据格式，如 JSON(MIME：application/json)、XML(MIME：text/xml)和 URL 编码数据(MIME：application/x-www-form-urlencoded)。Headers 指 HTTP 请求头，它同样是 Key-Value 型数据。因此，本文通过获取并分析 HTTP 访问日志来获取并解析 HTTP 请求。本文将 HTTP 日志中的 HTTP 请求解析为 Key-Value 型数据，并根据其来源将其分别存储到 param-key 列表、param-value 列表、headers-key 列表和 headers-value 列表，具体过程如图 1 所示。

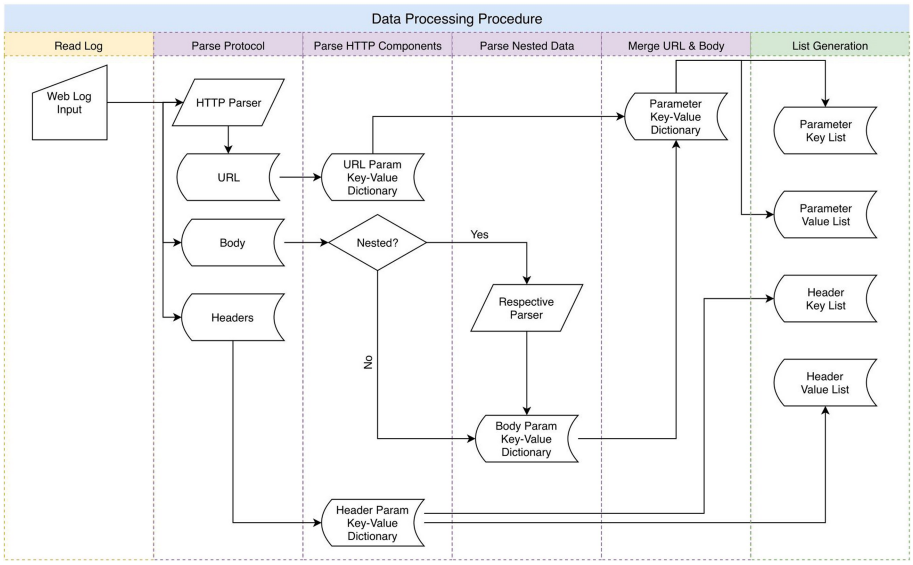

图 1 数据处理的流程

Fig. 1 Data processing procedure

上述过程完成后，对上述列表进行读取并将字符作为基本单元进行序列化。本文将所有字符划分为 5 种类型，如表 2 所列。

表 2 字符级序列的生成规则

Table 2 Character-level SEQ generation rule

SEQ	Description	Example
0,0,0,0,1	Unprintable Characters	N/A
0,0,0,1,0	Digit, Letter or Short Underline	1, A, _
0,0,1,0,0	Blank Characters	x0a, x00
0,1,0,0,0	Control Characters	x08
1,0,0,0,0	Special Characters	#, %, !

本文在对所有数据集完成数据预处理后，将其用于训练神经网络模型。

3.2 检测模型

本文最重要的目标是检测 Web 流量中的攻击载荷，通常来说，它们是一些被精心构造的代码片段。这些载荷内的字符之间存在关联关系，因此对其进行分析所需的技术与自然语言处理相近。RNN 具有的顺序依赖关系使其更为适合处理序列问题。而 LSTM 由于在 vanilla RNN 的基础上引入了门机制，有效地解决了 RNN 的短期依赖瓶颈问题，因此被广泛应用于如时间序列预测、自然语言处理、语音序列处理、流量分析等问题中。然而，传统的 RNN 一直存在梯度消失(Vanishing Gradient, VG)问题[19]，这导致 RNN 变得低效，即使是改进后的 RNN，如双向 RNN(Bidirectional RNN, BRNN)等，也依然存在该问题。受记忆细胞启发而实现的长短期记忆(Long Short-Term Memory, LSTM)[20]和门控循环单元(Gate Recurrent Unit, GRU)则解决了传统 RNN 的 VG 问题。

相比 LSTM，GRU 只具有 update 和 reset 两个门，参数更少且更易于收敛，因此更适合小样本集。本文方法通过 HTTP 日志持续地获取训练集，数据量十分充裕。针对数据量充裕这一现状，可知 LSTM 更适用于本文方法的应用场景。考虑到攻击载荷中的字符存在上下文关联性，本文最终选择双向 LSTM(BLSTM)网络作为构建检测模型的基础。

本文使用的 BLSTM 网络如图 2 所示，其包括输入为确定的向量标识的 BLSTM 层、Dense 层和 Softmax 层。BLSTM 层包含前向和后向两个方向，由多个 LSTM 单元组成。本文设计使用两个 BLSTM 层，每层包含 32 个 LSTM 单元，Dense 层用于降低 BLSTM 层的输出维度，Softmax 层输出二维的检测结果。

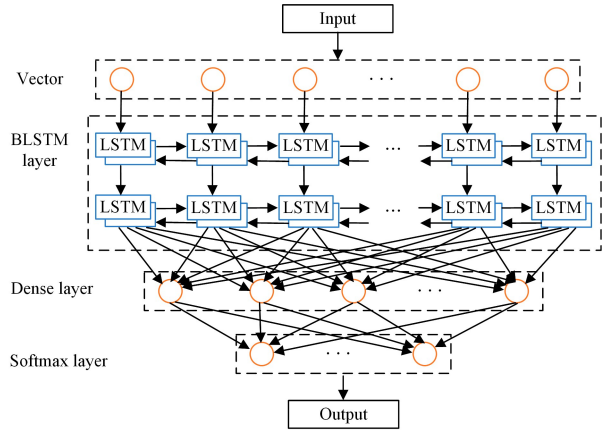

图 2 BLSTM 网络设计

Fig. 2 BLSTM network design

4 IoTGuardEye 的设计

4.1 架构设计

根据“生产环境线上业务优先”原则，为了达到最大限度地降低 IoTGuardEye 对线上业务的影响的目的，本文暂时不将 IoTGuardEye 作为 WAF 使用，而是将其应用于威胁感知与分析，用于捕获零日漏洞攻击载荷、发现攻击来源，在开销可接受的基础上缓解攻防不对称的情况。

如图 3 所示，IoTGuardEye 包含 3 个组成部分：运行在 IoT 服务器上的 Agent、用于管理调度的平台服务器和分析器集群。其中，Agent 用于在 IoT 服务器上收集 HTTP 日志并将其上传至平台服务器。平台服务器根据 HTTP 日志对请求进行解析，生成检测任务并将其推送给分析器集群。流量分析器集群按照任务队列顺序完成实际的检测工作，并将结果告知平台服务器，此后安全人员可以通过

平台服务器获取分析结果。

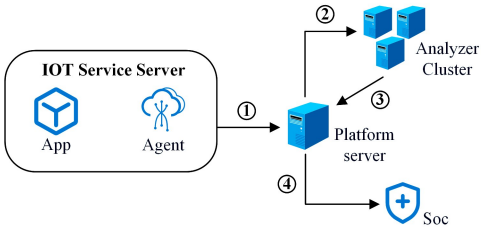

图 3　IoTGuardEye 架构设计

Fig. 3　Architecture design of IotGuardEye

4.2　流量分析器(Analyzer)

本文的流量分析器的工作流程如图 4 所示，它由一个模型选择器(Dispatcher)、一个标记器(Labeler)和多个 BLSTM 模型组成。本文定义产生数据交互、支撑物联网服务的 API 为“特殊 API”，这些 API 包含大量数据而且格式相对固定，需要被专门保护。根据这一定义，将所有检测模型划分为通用模型组和特殊 API 模型组，如表 3 所列。模型选择器根据请求路径的不同，为每个 HTTP 请求选择不同的检测模型组，并根据数据在 HTTP 请求中的位置(URI 参数、Body 或 Header)选择不同的模型分别进行检测，将检测结果传递至标记器。标记器的主要工作是与 WAF 联动，若某个载荷被模型识别为攻击载荷且该载荷不被 WAF 识别为攻击，则标记器会将其判断为“未知的攻击类型”，即零日漏洞攻击，这对于威胁感知具有重要意义。

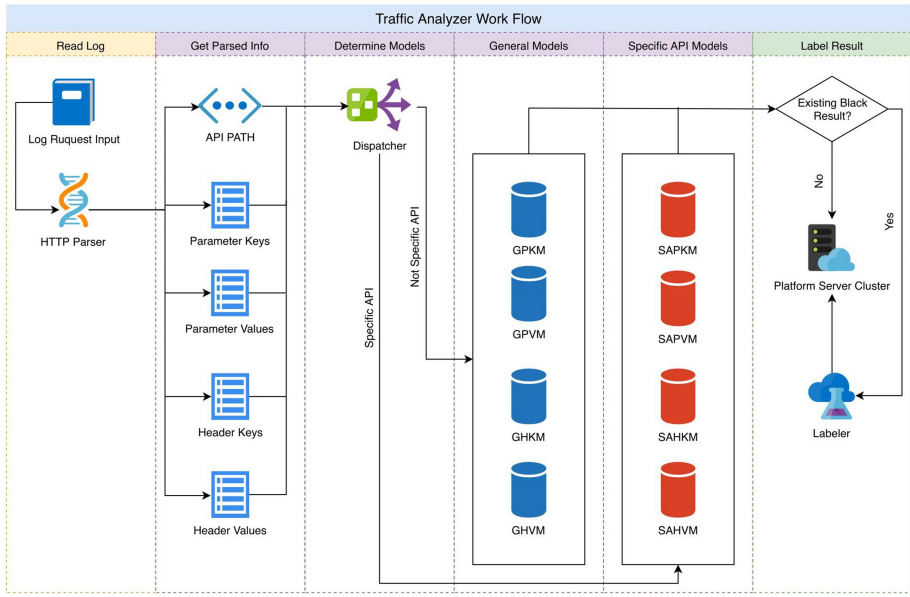

图 4　流量分析器流程

Fig. 4　Analyzer procedure

表 3　模型类型表

Table 3　Types of models

Name	Description
GPKM	General HTTP Parameter Key Model
GPVM	General HTTP Parameter Value Model
GHKM	General HTTP Header Key Model
GHVM	General HTTP HeaderValue Model
SAPKM	Specific API HTTP Parameter Key Model
SAPVM	Specific API HTTP Parameter Value Model
SAHKM	Specific API HTTP Header Key Model
SAHVM	Specific API HTTP Header Value Model

流量分析器的多模型设计与标记器使得 IoTGuardEye 在不减弱对普通流量的识别能力的基础上，可以达到更好的物联网服务 API 攻击检测效果，同时还可以有效发现 WAF 无法识别的未知攻击。

4.3　多模型的数据源

本文模型本质上是一个分类器，需要白样本和黑样本作为训练输入。就攻击类型而言，针对物联网 API 的攻击和针对传统 Web 的攻击并无本质区别，两者的攻击载荷具有共性。因此，无论是通用模型还是特殊 API 模型，其黑样本集均使用同样的攻击载荷样本集。

对于白样本集来说，通用模型的“通用”指模型并非是为某个 API 或某个服务专门设计的，而应尽可能地涵盖所有请求类型，因此训练通用模型应收集尽可能丰富的正常流量日志作为白样本集，不应局限于某条路径或某个 API。特殊 API 模型的主要特点是“针对性”，因此，针对每一个“特殊 API”，模型都需要收集尽可能多的该 API 的正常流量日志作为白样本集。

5　实验评估

本文在真实世界的数据集上进行效果评估，这些数据来自业界合作伙伴、部署于教育网的 Web 服务器和 WAF 日志、自行设计的概念验证(Proof of Concept，PoC)以及 IBM AppScan 等商用扫描器扫描 Web 服务器时产生的日志。

实验环境方面，本文使用一台带有 GPU 的服务器来模拟分析器集群，其配置参数如表 4 所列。本文的所有实验均在上述服务器中完成，包括 3.1 节所述的数据预处理部分。

表 4　实验环境

Table 4　Experiment environment

	Description
CPU	Intel Core i9-7900X @ 3.30 GHz (20 vCPU)
Memory	128 GB DDR4 3 000 MHz
GPU	NVIDIA GeForce RTX 2080Ti x 4
OS	Ubuntu 18.04.3 LTS

本节根据数据所处位置的不同将数据集划分为 Param 集和 Header 集；根据性质不同划分为 Normal 集和 Attack 集；根据流量类型的不同划分为通用集和特殊 API 集(SA 集)。因为本节仅验证模型效果，所以仅选择一个 API 作为特殊 API 进行日志提取，最终通过 HTTP 解析得到了 12 个训练集，如表 5 所列。

表 5 训练集

Table 5 Training datasets

Dataset	Count	Type	Models
S_{pNKey}	2032	Normal	GPKM
S_{pNVal}	3980	Normal	GPVM
S_{pAKey}	2004	Attack	GPKM/SAPKM
S_{pAVal}	4272	Attack	GPVM/SAPVM
S_{hNKey}	1000	Normal	GHKM
S_{hNVal}	1225	Normal	GHVM
S_{hAKey}	1003	Attack	GHKM/SAKM
S_{hAVal}	1276	Attack	GHVM/SAHVM
S_{SA_pNKey}	2000	Normal	SAPKM
S_{SA_pNVal}	4200	Normal	SAPVM
S_{SA_hNKey}	1000	Normal	SAHKM
S_{SA_hNVal}	1400	Normal	SAHVM

除上述数据集外，本文还用到了两个高质量的数据集，包括一个 XSS 数据集和一个 SQL 注入数据集。这两个数据集全部是由人工从近期的漏洞 PoC 和 WAF 日志中提取的，它们中的大部分攻击载荷是由人工精心构造的。本文将基于这两个数据集进行实验以作为对交叉验证的补充。

5.1 训练与评估

本文在表 5 的数据集上对每个模型进行了 50 轮训练，其结果如图 5 所示。

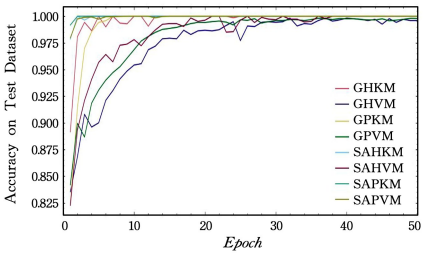

图 5 训练准确率变化

Fig. 5 Accuracy changes during the training period

结果表明，特殊 API 模型的收敛速度远快于通用模型；Key 相关模型的收敛速度远快于 Value 型数据的模型；SAPKM 与 SAPVM 的收敛速度相近。SAPKM 与 SAPVM 收敛速度相近的原因是对于特定的 API 而言，正常请求之间的 HTTP 参数差异是非常小的，这就使得异常流量变得更为显著，分类器可以更好、更快地学习到正常请求和恶意请求之间的差异，这也是本文设计特殊 API 模型的原因之一。

本文使用准确率(Accuracy)、召回率(Recall)和精确率(Precision)来评估本文模型。本文将攻击载荷标记为真(True)，正常载荷标记为假(False)，将识别为攻击载荷的模型分类结果标记为阳性(Positive)、识别为正常载荷的模型分类结果标记为阴性(Negative)。各指标的相关公式如下：

$$Accuracy=\frac{TP+TN}{TP+FN+FP+TN} \tag{1}$$

$$Recall=\frac{TP}{TP+FN} \tag{2}$$

$$Precision=\frac{TP}{TP+FP} \tag{3}$$

准确率指正确分类的样本数与所有样本数的比值，表示模型的综合检测能力。召回率指实际为阳性的样本中被预测为阳性样本的概率，它与模型对攻击载荷的检出率有关。精确率与正常有效载荷被错误地检测为攻击的可能性有关。交叉验证结果如表 6 所列。

表 6 交叉验证结果

Table 6 Cross-validation results

(单位：%)

Model	*Accuracy*	*Recall*	*Precision*
GPKM	100	100	100
GPVM	98.1	96.5	99.9
GHKM	100	100	100
GHVM	92.6	85.7	100
SAPKM	100	100	100
SAPVM	100	100	100
SAHKM	100	100	100
SAHVM	96.1	100	92.5

由表 6 可知，相比通用模型，特殊 API 模型更为有效；相比 Value 模型，Key 模型更为有效。本文认为这是因为特殊 API 和 Key 都具有格式相对固定、特殊字符相对较少的特点，攻击载荷和正常载荷的差异更为明显，因此神经网络可以更准确地学习到它们之间的差异，进而得到更好的分类效果。

5.2 系统级评估

本节不再使用“模型”作为基本单元，而是将 IoTGuardEye 作为一个整体与现有技术方案进行对比。本节在 XSS 和 SQLi 数据集上进行了实验。

参与测试的系统包括 IoTGuardEye 的通用模型、IoTGuardEye 的特殊 API 模型、OwlEye[21] 和天清 Web 安全网关(TQ-WAF)。OwlEye 是一个基于隐马尔可夫模型(Hidden Markov Model，HMM)实现的 Web 攻击检测系统；天清 Web 安全网关是由启明星辰公司研发的 WAF 系统，是中国市场占有率最高的 WAF 系统之一。

本节使用检出率(Detection Rate，DR)来评估待测系统的检测能力。若待测系统的检测结果为“恶意”或拦截了请求，则将其标记为阳性，反之标记为阴性。检出率的公式如下：

$$DR=\frac{TP}{TP+FN} \tag{4}$$

本实验通过提取数据集中的攻击载荷来构造 HTTP 请求，并将其发送给待测系统。实验结果如表 7 所列。

表 7 系统级评估结果

Table 7 System-level evaluation results

Type	Name	Positive	Count	DR/%
XSS	TQ-WAF	11722	11744	99.8
	OwlEye	4898	11744	41.7
	[General]	11340	11744	96.6
	[Special API]	11344	11744	96.6
SQLi	TQ-WAF	8765	18146	48.3
	OwlEye	16432	18146	90.6
	[General]	18023	18146	99.3
	[Special API]	18128	18146	99.9

实验过程中，由于 XSS 攻击载荷中的语义信息相对较少，而特殊符号和关键字具有较为明显的特征（包括尖括号、script 等）且变化较少，规则匹配（特别是正则表达式的匹配）相对容易，因此绝大部分 XSS 攻击载荷对 TQ-WAF 等知名 WAF 来说是“已知”的。强调零日漏洞攻击识别能力的 IoTGuardEye 在 XSS 数据集上的表现略逊于基于规则的 WAF，但与基于规则的传统 WAF 以及基于 HMM 的 OwlEye 相比，IoTGuardEye 具有明显的检出率优势，基于特殊 API 实现的模型效果也较通用模型更为优秀。综上，实验结果符合本文的设计预期和理论依据。

结束语　本文提出了面向物联网服务的 Web 攻击检测方案 IoTGuardEye，相比现有的技术方案，IoTGuardEye 不依赖于人工的规则设定，可以有效识别未知攻击载荷。实验结果表明，本文方法通过 BLSTM 实现了具有较好泛化能力的攻击检测功能，并通过模型选择器为不同的 API 定制了不同的检测模型，有效增强了 BLSTM 模型在物联网 API 流量上的检测效果。未来，将尝试降低该方案的开销，在成本可控的情况下实现阻断功能，达到 WAF 的效果。

致谢　感谢浙江大学的于清晨博士、兰州大学张文强同学对本文撰写的帮助。感谢海丰科技、金睛云华和盛邦安全对本文数据集工作的支持。

参 考 文 献

[1] ALFONSO V, JAMES F H, HUNG L H, et al. Predicts 2015: The Internet of Things[EB/OL]. (2014-12-30) [2020-07-28]. https://www.gartner.com/doc/2952822/predicts-internet-things.

[2] ALAN N, ALEX S. Internet security threat report[EB/OL]. (2019-02) [2020-07-28]. https://symantec.broadcom.com/symc-istr-v24-2019-6819.

[3] WAKEFIELDR L. Computer monitoring and surveillance [J]. The CPA Journal, 2004, 74(7): 52.

[4] DENNING D E. An Intrusion-Detection Model[J]. IEEE Transactions on Software Engineering, 1987, SE-13(2): 222-232.

[5] YU F, CHEN Z F, DIAO Y L, et al. Fast and memory-efficient regular expression matching for deep packet inspection[C]// 2006 ACM/IEEE symposium on Architecture for networking and communications systems (ANCS'06). 2006: 93-102.

[6] ROESCH M. Snort - Lightweight Intrusion Detection for Networks[C]// In Proceedings of the 13th USENIX conference on System administration (LISA '99). Association, USA, 1999: 229-238.

[7] KRUEGEL C, TOTH T. Using Decision Trees to Improve Signature-Based Intrusion Detection[C]// Recent Advances in Intrusion Detection (RAID 2003). Lecture Notes in Computer Science, 2003: 173-191.

[8] CHEN W H, HSU S H, SHEN H P. Application of SVM and ANN for intrusion detection[J]. Computers & Operations Research, 2005, 32(10): 2617-2634.

[9] LIANG J, ZHAO W, YE W. Anomaly-Based Web Attack Detection: A Deep Learning Approach[C]// The 2017 VI International Conference on Network, Communication and Computing (ICNCC 2017). 2017: 80-85.

[10] SAXE J, BERLIN K. Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features[C]// 2015 10th International Conference on Malicious and Unwanted Software (MALWARE). Fajardo, 2015: 11-20.

[11] TIAN Z, LUO C, QIU J, et al. A Distributed Deep Learning System for Web Attack Detection on Edge Devices[J]. IEEE Transactions on Industrial Informatics, 2020, 16(3): 1963-1971.

[12] TAMA B A, NKENYEREYE L, ISLAM S M R, et al. An Enhanced Anomaly Detection in Web Traffic Using a Stack of Classifier Ensemble[J]. IEEE Access, 2020, 8: 24120-24134.

[13] MOHAMMADI S, NAMADCHIAN A. Anomaly-based Web Attack Detection: The Application of Deep Neural Network Seq2Seq With Attention Mechanism[J]. The ISC International Journal of Information Security, 2020, 12(1): 44-54.

[14] DU M, LI F, ZHENG G, et al. DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning[C]// The 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS'17). 2017: 1285-1298.

[15] JIANG F, FU Y, GUPTAB B, et al. Deep Learning based Multichannel intelligent attack detection for Data Security[J]. IEEE Transactions on Sustainable Computing, 2020, 5(2): 204-212.

[16] JIN X, CUI B, YANG J, et al. Payload-Based Web Attack Detection Using Deep Neural Network[C]// Advances on Broad-Band Wireless Computing, Communication and Applications (BWCCA 2017). Lecture Notes on Data Engineering and Communications Technologies, 2018: 482-488.

[17] SKARUZ J, SEREDYNSKI F. Recurrent neural networks towards detection of SQL attacks[C]// 2007 IEEE International Parallel and Distributed Processing Symposium. Rome, 2007: 1-8.

[18] LIU H Y, LANG B, LIU M, et al. CNN and RNN based payload classification methods for attack detection[J]. Knowledge-Based Systems, 2019, 163(1): 332-341.

[19] LI Z, ZOU D, XU S, et al. VulDeePecker: A Deep Learning-Based System for Vulnerability Detection[C]// Network and Distributed System Security Symposium. 2018: 23158.

[20] HOCHREITER S, SCHMIDHUBER J. Long Short-Term Memory[J]. Neural Computation, 1997, 9(8): 1735-1780.

[21] YONG B, LIU X, YU Q, et al. Malicious Web traffic detection for Internet of Things environments[J]. Computers & Electrical Engineering, 2019, 77: 260-272.

LIU Xin, born in 1995, Ph.D student. His main research interests include web security, IoT security and blockchain security.

ZHOU Rui, born in 1981, associate professor. His main research interests include distributed systems, embedded systems and machine learning.