

融合时间卷积网络与注意力机制的APT攻击检测

崔帅¹, 王金双^{1,2}, 赵宁¹, 瞿康健¹

(1. 陆军工程大学指挥控制工程学院; 2. 先进通信技术国家级重点实验室, 江苏, 南京 210007)

摘要: 高级持续性威胁(APT)以其持续时间长和隐蔽性强的特点, 给检测技术带来了严峻挑战。近年来, 研究者们将溯源图应用于APT攻击检测, 通过节点和边的关系有效地建模攻击行为特征。然而, 现有的基于溯源图的检测方法在长期因果关系建模和语义信息利用方面存在局限性, 因此, 提出了一种融合时间卷积网络和注意力机制的检测方法STANet。首先, 采用Word2Vec联合位置编码捕获关键语义属性与事件时序特征, 生成语义增强的特征向量序列; 其次, 利用时间卷积网络中的空洞因果卷积提取持续攻击的因果关系特征, 并引入注意力机制捕捉关键特征, 从而有效地识别恶意节点。实验结果表明, 在10个公开的数据集上, STANet实现了平均95.39%的精确度和98.69%的召回率, 显著优于现有基线方法。

关键词: 高级持续性威胁; 溯源图; 异常检测; Word2Vec; 时间卷积网络; 注意力机制

DOI: 10.11907/tjdc.251158

扫描二维码阅读全文:



中图分类号: TP18

文献标识码: A

文章编号: 1672-7800(2026)006-0142-08

An APT Attack Detection Method Combining Temporal Convolutional Network and Attention Mechanism

CUI Shuai¹, WANG Jinshuang^{1,2}, ZHAO Ning¹, QU Kangjian¹

(1. Institute of Command Control Engineering, Army Engineering University;

2. State Key Laboratory of Advanced Communication Technology, Nanjing 210007, China)

Abstract: Advanced Persistent Threats (APTs) are characterized by their prolonged duration and highly stealthy nature, therefore, it is particularly challenging to detect them with conventional methods. Recent research has shown growing interest in detection approaches based on provenance graphs, which can effectively model attack behavior characteristics with nodes and edges. Existing methods utilizing provenance graphs still face challenges in capturing long-term casual dependencies and fully leveraging semantic information. This paper introduces STANet, an APT detection method that integrates temporal convolutional networks with attention mechanisms. STANet employs a Word2Vec-based encoder to extract critical semantic attributes and temporal ordering of events within the provenance graph, thereby constructing semantically enhanced vector representations. The method further employs causal dilated convolutions within TCNs to effectively extract casual relationships features of persistent attack campaigns, while the attention modules dynamically prioritize mission-critical features across different time steps. This dual mechanism enables precise identification of malicious nodes within provenance graphs. Experimental evaluations on ten public datasets demonstrate that STANet achieves an average precision of 95.39% and recall of 98.69%, significantly outperform existing baseline methods.

Key Words: advanced persistent threat; provenance graph; attack detection; Word2Vec; temporal convolutional network; attention mechanism

0 引言

高级持续性威胁(Advanced Persistent Threat, APT)具有长期潜伏性、隐蔽性强和针对性高等突出特点, 这使得

对其进行精确检测面临两大核心挑战: ①传统方法依赖于人工规则, 难以从海量日志中提取细粒度攻击步骤; ②现有基于系统日志的方法忽视了事件间语义关联, 导致攻击链检测能力不足。近年来, 基于溯源图(Provenance Graph)的检测方法通过构建系统实体交互的有向图, 有效

收稿日期: 2025-04-15

作者简介: 崔帅(1998-), 女, 陆军工程大学指挥控制工程学院硕士研究生, 研究方向为系统安全; 王金双(1978-), 男, 博士, 陆军工程大学指挥控制工程学院副教授、硕士生导师, 研究方向为系统安全; 赵宁(1993-), 女, CCF学生会会员, 陆军工程大学指挥控制工程学院硕士研究生, 研究方向为系统安全; 瞿康健(1999-), 男, 陆军工程大学指挥控制工程学院硕士研究生, 研究方向为信息安全。本文通讯作者: 王金双。

整合语义信息与历史关联,成为研究热点^[1-3]。然而,在复杂的安全环境中,当前基于溯源图的检测方法仍面临关键语义信息缺失和长期因果关系捕获不足的局限性。因此,在特征提取和关键语义信息学习方面仍需进一步优化,增强对溯源图的语义和上下文信息的捕获,从而提高检测精度。

1 相关研究

溯源图是对系统审计日志的抽象处理,其节点表示系统实体,边表示实体之间的交互关系,能够将因果相关的攻击事件直接关联起来。图 1 展示了一个真实 APT 攻击的溯源图实例。与原始系统审计数据相比,溯源图不仅提供丰富的语义信息,还具备强大的历史关联能力。

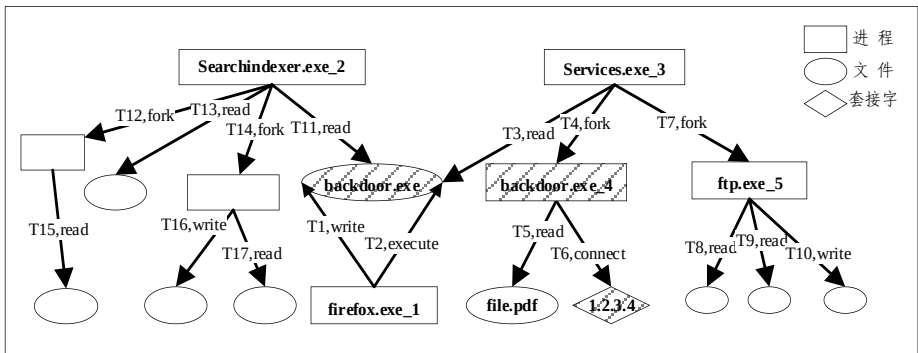


Fig. 1 An example of provenance graph

图 1 溯源图实例

随着深度学习技术的发展,研究者将其引入基于溯源图的 APT 攻击检测中,利用深度学习自动挖掘海量数据中隐藏模式的能力,从而提高异常检测效率和自动化程度。基于溯源图的 APT 检测方法总结,如表 1 所示。

由表 1 可知,基于序列表示学习的检测方法中,UNICORN 利用图核算法构建系统行为直方图,并通过聚类实现异常检测^[10];ATLAS 利用因果分析和序列建模提取攻击和非攻击模式^[11];SeqNet 采用门控循环单元(Gate Recurrent Unit, GRU)模型提取长期依赖特征^[12];PROGRAPHER

Table 1 Summary of provenance graph-based APT detection methods

表 1 基于溯源图的 APT 检测方法总结

分类	名称	检测方法	攻击场景图
基于序列表示学习的检测方法	UNICORN ^[10]	图核+聚类	否
	ATLAS ^[11]	词嵌入+LSTM	是
	SeqNet ^[12]	GRU+局部注意力	否
	PROGRAPHER ^[13]	Graph2Vec+TextRCNN	是
	DACGN ^[14]	DNN+注意力机制+GRU	否
	Pro2vec ^[15]	图核	否
基于图表示学习的检测方法	THREATTRACE ^[16]	GraphSAGE+异常分类	否
	Kairos ^[17]	TGN+社区发现	是
	Magic ^[18]	图自编码器+GAT	是

目前,溯源图被广泛用于 APT 的攻击检测。SLEUTH 通过可信任度和机密性标签传播构建异常检测模型^[4], MORSE 提出了标签衰减策略来缓解标签污染问题^[5]。然而,随着图规模的增长,基于标签传播的攻击检测方法难以避免“依赖爆炸”问题。HOLMES 基于 ATT&CK 框架预定义匹配规则,通过构建高级场景图(High-level Scenario Graph, HSG)将低层系统调用映射到高级杀伤链^[6],以此识别潜在 APT 攻击^[7]。POIROT 通过网络威胁情报(Cyber Threat Intelligence, CTI)驱动的图模式匹配识别攻击^[8]。MARLIN 采用流式图对齐框架,通过预定义查询图匹配攻击模式,结合标签传播进行异常检测^[9]。然而,这类基于规则的攻击检测方法通常需要依赖外部网络攻防知识或专家经验来制定匹配规则,因此难以快速更新以适应新型威胁。

使用 graph2vec 图嵌入技术为溯源图快照中的每个节点生成根子图(RSG, Rooted Subgraph)的低维表示,并利用 TextRCNN 模型检测快照偏差^[13];DACGN 结合深度神经网络(DNN)和注意力机制增强模型对语义的捕获能力^[14];Pro2vec 通过标签感知后向游走捕获节点的历史行为,生成节点标签直方图^[15]。基于图表示学习的检测方法中,THREATTRACE 利用改进的图神经网络(GraphSAGE)实现节点级别的 APT 攻击检测^[16]。Kairos 通过编码器-解码器架构学习溯源图结构的动态变化,利用系统实体的历史行为模式量化溯源图中系统事件的异常程度^[17]。Magic 利用图掩码自编码器对溯源图进行特征重构与邻域采样结构重建,结合 K-D 树优化的 K 近邻异常检测算法进行异常检测^[18]。

上述基于深度学习技术的检测方法存在以下局限性:①关键实体语义(如进程名、文件路径以及 IP 地址等)利用不充分;②基于图表示学习的检测方法随着图规模增大,其可扩展性面临挑战,导致对长期依赖关系捕获受限;③基于序列表示学习的检测方法,由于模型限制导致对长期攻击特征学习能力不足。

综上,本文通过融合 Word2Vec^[19]、时间卷积网络(Temporal Convolutional Network, TCN)^[20]和注意力机制^[21],提出了一种名为 STANet(Semantic-Temporal Attention Net-

work)的基于溯源图的APT攻击检测方法,旨在提高对隐蔽APT攻击的识别能力。本文主要贡献如下:①结合使用Word2Vec和位置编码,通过位置编码对Word2Vec生成的节点语义特征进行时序增强,获取语义增强的特征向量序列;②将TCN用于APT攻击检测,利用因果卷积(Causal Convolution)、空洞卷积(Dilated Convolution)和残差模块(Residual Block),提高对长周期、多阶段攻击行为的时序关联,解决传统CNN在时序建模中存在的因果性缺失和感受

野受限的问题,同时避免RNN的串行计算瓶颈;③引入注意力机制增强对关键特征的提取,以提高异常检测能力。

2 STANet方法

STANet(Semantic-Temporal Attention Network)实现过程可分为3个关键阶段:数据预处理、序列学习和异常检测,具体如图2所示。

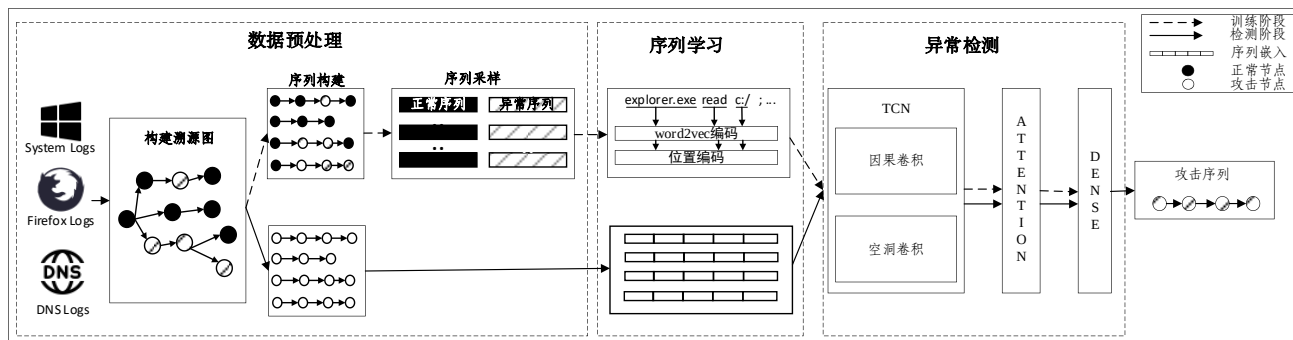


Fig. 2 The framework of STANet

图2 STANet 框架图

2.1 数据预处理

数据预处理旨在将溯源图转化为事件序列,为后续的异常检测提供结构化的输入。该阶段依次进行溯源图构建和序列构建。

(1)溯源图构建。STANet通过对系统审计日志(如系统事件日志、DNS服务器日志和浏览器日志)进行解析,构建有向、带时间戳的溯源图。在溯源图中,每个系统事件被表示为一条边,其中,源节点表示为事件的主体(如进程),目的节点表示为事件的对象(如文件)。为了全面描述系统的动态行为,STANet框架考虑了进程、文件和网络套接字这三类节点,并依据其属性进行了细粒度分类。节点和边的具体类型如表2所示。

Table 2 Types of nodes and edges

表2 节点和边的类型

分类	类型
节点	process, system_process, user_process, programfiles_process
	file, system_file, user_file, programfiles_file
	IP_Address, domain_name, web_object, connection, session
边	read, write, delete, execute, fork, connect, bind, resolve, web_request, refer, send

(2)序列构建。STANet在训练阶段基于攻击真值标注,从溯源图中提取攻击序列和正常序列进行模型训练,以学习正常和攻击行为的模式差异。首先,从溯源图中获取所有已知的攻击节点集合,并基于这些节点构建包含2个攻击节点及以上的子集。对于每个子集,提取其1跳邻

域内的节点及边,构建局部邻域图。其次,从邻域图中查找源节点或目的节点属于当前攻击子集的事件,并将其标记为攻击事件。最后,将提取的事件按照时间戳排序,形成一个有序的时间序列。若该序列只含有攻击事件并且包含子集中所有节点的攻击事件,则将其标记为攻击序列。

为了更准确地学习正常行为与攻击行为之间的差异,STANet在每个攻击子集的1跳邻域内随机选择一个正常节点加入。采用与提取攻击序列类似的方法,从邻域图中提取相关节点的事件序列。如果提取的序列与攻击序列不匹配,则将其标记为正常序列。

为了解决训练数据集中类别不平衡问题,STANet采用了一种两步策略:首先,采用Levenshtein Distance对正常序列进行欠采样,对于每个序列,当其相似性超过某一特定阈值时,STANet会移除这些序列,以减少数据冗余并提高数据集的质量^[22];其次,采用过采样技术对攻击序列进行随机变异,逐步增加样本数量,直至攻击序列与正常序列数量平衡。实验结果表明,当相似性阈值设置为0.8时,能够取得较为理想的欠采样效果,有效过滤掉高度相似及冗余的正常序列。

2.2 序列学习

序列学习阶段使用语义编码器分别对节点属性和位置进行编码,以生成语义增强的特征向量。

2.2.1 节点属性编码

系统审计日志中实体具有丰富的属性信息,如进程名、文件路径以及网络IP地址等。后文中实体(Entity)和节点(Node)这两个术语会交替使用,在描述溯源图的具体结构时,会使用节点来强调其在图中的位置和角色,而其

他上下文中可能会用实体来表示相同的概念。为了充分利用这些属性,首先需要对其进行编码。然而,传统的one-hot编码方式会产生高维稀疏矩阵,并且忽略每个单词周围的上下文关系,无法充分表达语义信息。

Word2Vec是一种基于神经网络的词嵌入方法,专为学习单词的低维稠密向量表示而设计^[19]。该模型通过预测单词的上下文或根据上下文预测单词,捕捉单词之间的语义关系。因此,STANet采用Word2Vec模型,将每个实体的属性转换为稠密向量空间表示。STANet选取的3种实体,其属性中包含进程名、文件路径、网络IP地址和端口号,有助于生成语义丰富的特征向量。将提取的序列作为句子,使用基于所有序列训练的Word2Vec模型将其编码为固定长度向量。Word2Vec编码后,语义相似的单词处于相近的位置。通过这种方法不仅可以学习单词之间的上下文信息,还能够学习攻击行为和正常行为之间的差异。

2.2.2 位置编码

由于APT攻击检测高度依赖事件时序关系,而Word2Vec生成的词嵌入缺乏时序信息,因此,采用位置编码将事件的时序特征注入嵌入表示^[21]。位置编码向量由式(1)生成:

$$PE_{(pos, 2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \quad (1)$$

$$PE_{(pos, 2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

其中, pos 表示事件在序列中的位置, d 是嵌入维度。随后,逐元素将位置编码向量与Word2Vec嵌入向量相加,得到第 i 个事件的最终嵌入表示,如式(2)所示:

$$Emb_i = Emb_i + PE_i \quad (2)$$

算法1概述了生成特征向量的过程,此算法结合Word2Vec生成的语义向量和位置编码的时序信息,不仅可以捕获节点之间的语义信息,还可以捕获序列中节点的顺序信息。这种丰富的特征表示使得后续的异常检测模型能够更好地理解上下文信息。

算法1 语义增强编码

输入:序列 S ; 训练好的Word2Vec模型 $w2v$

输出:句子编码向量 V

1. $D \leftarrow \text{list}([])$ /*遍历序列 S 中的每个节点*/
2. FOR $N \in S$ DO /*获取节点 N 上的交互操作 O */
3. $O = \text{GetOperation}(N)$ /*获取节点 N 的属性,包含进程名、文件路径和IP地址等信息*/
4. $A = \text{GetNodeAttributes}(N)$ /*将操作词 O 和属性词 A 记录到列表 D */
5. $D.append(O)$
6. $D.append(A)$
7. END
8. $V \leftarrow \text{list}([])$ /*初始化一个空列表 V 用于存储特征向量*/
9. FOR $w \in D$ DO /*遍历列表 D 中的单词*/

10. $E = w2v(w)$ /*生成Word2Vec嵌入*/

11. $V.append(E)$

12. END

13. $P = \text{GetPositionalEncodingVector}(\text{len}(V))$ /*生成位置编码*/

14. $V \leftarrow V + P$ /*叠加Word2Vec嵌入向量与位置权重*/

15. $V \leftarrow V.mean()$ /*平均池化,获得整个序列的特征向量*/

16. RETURN V

2.3 异常检测

生成特征向量序列之后,STANet采用时间卷积网络(TCN)与注意力机制的混合架构,从攻击和正常序列中构建检测模型。模型训练阶段,将训练集中的特征向量序列作为输入。给定一组序列 $S_i = \{x_1, x_2, \dots, x_T\}$ 时,使用TCN来获取输入序列中每个 x_i 的隐藏表示,其空洞因果卷积公式如式(3)所示:

$$h_i = \phi\left(\sum_j w_j x_{i-j} + b\right) \quad (3)$$

其中, w_j 为卷积核的权重, b 为偏置向量, ϕ 是激活函数。随后,使用残差连接获得式(4):

$$y_i = h_i + x_i \quad (4)$$

经过多层卷积输出逐层传递,最终获得 S_i 的特征向量 $X = [y_1^{(L)}, y_2^{(L)}, \dots, y_T^{(L)}]$, 其中, L 为TCN的层数。

注意力机制通过注意力权重矩阵使模型能够自动区分序列中数据的重要性,并且能够直接建模任意距离的时间步关联。而TCN由于靠卷积核大小限制了感受野,注意力层能够弥补TCN在长序列建模中的不足。为此,将TCN处理后生成的特征向量 X 输入注意力机制层,令输入的矩阵 X 与随机初始化矩阵 W^Q 、 W^K 、 W^V 分别相乘,获得查询(Q)、键(K)、值(V)矩阵,其计算公式如式(5)~式(7)所示:

$$Q = X \times W^Q \quad (5)$$

$$K = X \times W^K \quad (6)$$

$$V = X \times W^V \quad (7)$$

通过计算得到最终特征向量 Z , 其计算公式如式(8)所示:

$$Z = \text{soft max}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (8)$$

注意力层引入的过程使得模型能够计算时间步间的相似性,捕获局部与全局的复杂模式,从而提升特征表现能力。STANet使用一个全连接层来获取每个序列的最终表示,由Sigmoid函数对特征向量进行激活,最终得到属于攻击相关序列的可能性。通过交叉熵损失函数对模型预测与真实标签间的差异进行精确衡量,其定义如式(9)所示:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \quad (9)$$

其中, y_i 是真实标签, $\hat{y}_i$ 是模型预测的概率。

在测试阶段,本文采用与训练阶段一致的方法来提取序列,并将测试序列输入到异常识别模型中。通过将模型输出与实际标签进行对比,若两者差距超过预定义阈值,则判定该序列为异常,并将该序列中的节点标记为异常节点。

3 实验评估

通过 10 个数据集验证 STANet 在 APT 攻击检测上的有效性,选择 2 个基线模型 ATLAS^[11]和 DACGN^[14]以及 2 个替换模块方法进行比较。

3.1 实验设置

3.1.1 数据集

ATLAS 数据集 (https://github.com/purselab/ATLAS/tree/main/raw_logs)是由 Alsaheel 等^[11]基于真实 APT 报告

实施模拟攻击而生成的公开可用的审计日志。该数据集在隔离网络环境中模拟了为期 10 天的企业环境。为了增加真实性,在用户 8 h 工作时间内随意执行攻击活动。攻击者采用不同的攻击战术,如钓鱼邮件、注入攻击以及横向移动等。同时,收集日志中也包含大量的正常行为,如浏览各种网站、打开各种应用程序(阅读邮件、下载附件等)以及远程服务器交互等。表 3 介绍了该数据集的详细信息,包含攻击使用的 CVE、攻击战术、日志大小、日志类型以及节点和事件的数量。平均每项攻击产生 20 088 个实体,平均事件数量约为 249 K。数据集 S1-S4 是在单个主机上进行的攻击,M1-M6 是在多个主机上进行的攻击。对于每个多主机攻击场景,本文在两台主机上模拟了攻击与正常行为,其中,第二台主机被设定为横向移动的目标。所有攻击均在 Windows7 32 位虚拟机上开发并执行。

Table 3 Overview of the ATLAS datasets

表 3 ATLAS 数据集概述

数据集	CVE	攻击战术							大小/MB	日志类型/%			数量	
		PL	PA	INJ	IG	BD	LM	DE		System	Web	DNS	实体	事件
S1	2015-5122	√		√	√	√		√	381	97.11	2.24	0.65	7 468	95.0K
S2	2015-3105	√		√	√	√		√	990	98.58	1.09	0.33	34 021	397.9K
S3	2017-11882		√	√	√	√		√	521	96.82	2.43	0.75	8 998	128.3K
S4	2017-0199		√	√	√	√		√	448	97.08	2.24	0.68	13 037	125.6K
M1	2015-5122	√		√	√	√	√	√	851.3	96.89	1.32	1.32	17 599	251.6K
M2	2015-5119	√		√	√	√	√	√	819.9	97.39	1.36	1.25	24 496	284.3K
M3	2015-3105	√		√	√	√	√	√	496.7	99.11	0.52	0.37	24 481	334.1K
M4	2018-8174		√	√	√	√	√	√	653.6	98.14	1.24	0.62	15 409	258.7K
M5	2017-0199	√		√	√	√	√	√	878	98.63	0.97	0.40	35 709	258.7K
M6	2017-11882		√	√	√	√	√	√	725	98.31	0.96	0.73	19 666	354.0K
平均	-	-	-	-	-	-	-	-	676.5	97.76	1.46	0.73	20088	249K

注:PL:钓鱼邮件链接 PA:钓鱼邮件附件 INJ:注入攻击 IG:信息搜集 BD:后门攻击 LM:横向移动 DE:数据泄露

3.1.2 实验环境及参数

STANet 模型基于 Keras 环境(以 TensorFlow 为后端)实现。所有模型均在 Python 3.7 环境下完成,并采用 Gensim 库实现了 Word2Vec 模块。对于模型使用的参数,本文将词嵌入维度设置为 128,序列长度设置为 400,卷积维度设置为 64,卷积核数为 5,阈值设置为 0.5,学习率设置为 0.001,使用 Adam 作为优化器。所有实验均在一台运行 Ubuntu20.04 LTS 的服务器上进行,其配置如下: Intel® Core™ i5-9600K CPU@3.70GHz×6 处理器、64 G 内存、1 TB SSD 和 NVIDIA GTX1080Ti 11 G。

3.1.3 评价指标

STANet 是一种分类任务,如果一个序列符合攻击序列的条件,则该序列被识别为攻击序列。序列中包含的节点为攻击节点,否则为正常序列,其节点也为正常节点。在这种情况下,正确检测到攻击序列时,计算为真阳性(TP);未检测到攻击时,计算为假阴性(FN)。同理,在正常数据上定义真阴性(TN)和假阳性(FP)。分类任务常用的评价

指标通常包括准确率(Precision)、召回率(Recall)和 F1 值(F1-Score),其计算方式如式(10)~(12)所示:

$$Precision = \frac{TP}{TP + FP} \quad (10)$$

$$Recall = \frac{TP}{TP + FN} \quad (11)$$

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (12)$$

3.2 实验有效性分析

对于每次 APT 攻击,攻击开始前已知攻击实体的真实情况,如在攻击中设置的恶意 IP 地址。为了评估 STANet 在检测 APT 攻击方面的性能,在 APT 攻击数据集上训练模型时,数据集的 80% 用于训练,20% 用于验证。例如,对于单主机 APT 攻击 S1,使用其他 3 个单主机数据集(S2、S3 和 S4)来训练模型;对于多主机 APT 攻击 M2,使用其他 5 个多主机数据集(M1、M3、M4、M5 和 M6)来训练模型。

模型训练完成后,STANet 开始进行检测。首先,根据警报实体从溯源图中提取序列;其次,利用训练好的模型

预测序列是否为攻击序列;最后,将预测结果与真值进行对比,判断警报节点是否为攻击节点。

3.2.1 模型检测效果分析

为了验证 STANet 模型在识别 APT 攻击节点方面的有效性,在每个数据集上进行了系统性实验分析。表 4 汇总了模型在检测 APT 攻击节点方面的性能指标(精度、召回率及 F1 值)。

Table 4 Experimental results of STANet

表 4 STANet 的实验结果

数据集	警报实体	攻击节点的调查结果						
		TP	TN	FP	FN	Precision/%	Recall/%	F1-score/%
S1	恶意主机	22	7 445	0	0	100.00	100.00	100.00
S2	泄露文件	12	34 010	0	0	100.00	100.00	100.00
S3	恶意主机	27	8 970	0	1	100.00	96.43	98.18
S4	泄露文件	21	13 013	3	0	87.50	100.00	93.33
M1	泄露文件	28	17 563	2	0	93.33	100.00	96.55
M2	泄露文件	37	24 447	3	0	92.50	100.00	96.10
M3	恶意文件	36	24 423	1	0	97.30	100.00	98.63
M4	恶意文件	26	15 378	0	2	100.00	92.86	96.30
M5	恶意主机	32	35 669	2	0	94.17	100.00	96.99
M6	恶意主机	41	19 575	5	1	89.13	97.61	93.17
平均	-	28.20	20 049.30	1.60	0.40	95.39	98.69	96.93

由表 4 可知,STANet 在识别异常节点方面展现出了优异的性能,平均精度达到 95.39%,召回率为 98.69%,F1 值为 96.93%。实验结果表明,此模型能够高效且准确地识别出大多数攻击实体,同时,保持较低的误报率。

通过对系统事件与表 4 数据的综合分析发现,大多数误报都源于 IP 地址的错误分类。例如,在 M5 和 M6 中,大多数误报源于那些与命令与控制(C2)服务器同时期活跃的正常 IP 地址。这表明,尽管这些 IP 地址参与的是正常网络活动,但其行为特征与恶意活动存在相似性,从而导致误报。通过安全分析人员对同一特定 IP 进行人工审查过滤,可以减少此类误报情况。

3.2.2 与基线方法对比分析

为了验证 STANet 方法的有效性和效率,实验中选择了 ATLAS^[11]和 DACGN^[14]2 个基线方法进行对比实验。由于之前基于溯源图的 APT 攻击检测方法很少完全使用系统日志作为原始数据,ATLAS 提出了一种模块替换方法,以原方法为基准,用某个方法替换模型的对应模块,观察其对精度、召回率、F1 值等指标的影响。本文沿用这个方法,并选择了图遍历(Graph-traversal)和 one-hot 编码 2 个替换模块。STANet 与基线方法的对比实验结果如表 5 所示。

在图遍历方法中,使用基于 W3C 的溯源数据构建的有向非循环图执行前向和后向追踪。首先,从警报实体开始后向追踪,在攻击的根本原因(如恶意 URL)处停止;其次,从警报实体开始前向追踪寻找攻击目标(如泄露的机密文档);最后,根据前向跟踪和后向跟踪获得的实体和事件,

Table 5 Comparison results between STANet and baseline methods

表 5 STANet 与基线方法的对比实验结果

方法	Precision/%	Recall/%	F1-score/%
图遍历	21.82	100.00	35.82
One-hot 编码	90.09	77.99	83.59
ATLAS ^[11]	91.06	97.29	93.76
DACGN ^[14]	93.18	97.94	95.35
STANet	95.39	98.69	96.93

识别攻击节点。此方法的结果如表 5 所示,召回率为 100%,但是精度只有 21.82%。产生这种结果的原因是溯源图的“依赖爆炸”,该问题在图遍历的时候引入了大量的正常事件作为错误攻击依赖。

为了验证 Word2Vec 的有效性,使用 one-hot 编码替换 Word2Vec,其检测精度、召回率和 F1 值分别降低到 90.09%、77.99% 和 83.59%。这主要由于 one-hot 编码忽略了单词之间的语义关联。结果表明,Word2Vec 可以使模型学习到更多区分攻击序列和正常序列的特征。

在 APT 攻击的节点检测中,ATLAS 的精度、召回率和 F1 值与 STANet 相比,分别降低了 4.33%、1.4% 和 3.17%;DACGN 的精度、召回率和 F1 值与 STANet 相比,分别降低了 2.21%、0.75% 和 1.58%。其原因如下:①Word2Vec 对节点属性进行了细粒度学习,有效利用了上下文的语义信息,增强了节点的语义特征;②时间卷积网络和注意力机制提高了长期因果关系的捕获能力,提供了额外的语义信息去识别攻击和正常序列之间的差异。

3.2.3 参数有效性分析

在不同嵌入维度和序列长度的参数设置下,相应的检测效果对比结果如图 3 所示。

(1)嵌入维度。在嵌入过程中,节点特征被嵌入并映射到低维向量上,同时尽可能多地保留其原始特征属性。如图 3(a)所示,随着嵌入维数的增加,检测指标的得到了显著提高,说明嵌入维度越大,可以捕获的信息更多。然而,维度过大会导致特征稀疏性,影响检测效果。当维数设置为 128 时,模型能够达到最佳检测效果。

(2)序列长度。序列长度是指一个序列中实体和操作的总数量。STANet 在系统构建序列的过程中,可能会生成不同长度的序列,因为每个序列由不同数量的事件组成。为确保高效处理,系统对所有序列执行了统一长度处理。系统日志记录了大量的正常活动,其中频繁出现的正常活动往往构成较长的序列,而攻击活动则通常以短序列的形式出现。此外,利用攻击节点提取攻击序列时,长序攻击序列也被其子序完全表征。由图 3(b)可知,当序列长度超过 400 时,检测精度略有下降,原因在于长序列中正常事件噪声增加。实验表明,当序列长度为 400 时检测效果最佳。序列长度与 TCN 的感受野直接相关。为确保感受野能覆盖整个序列,本文采用网格搜索法来寻找 TCN 的最优参数组合。

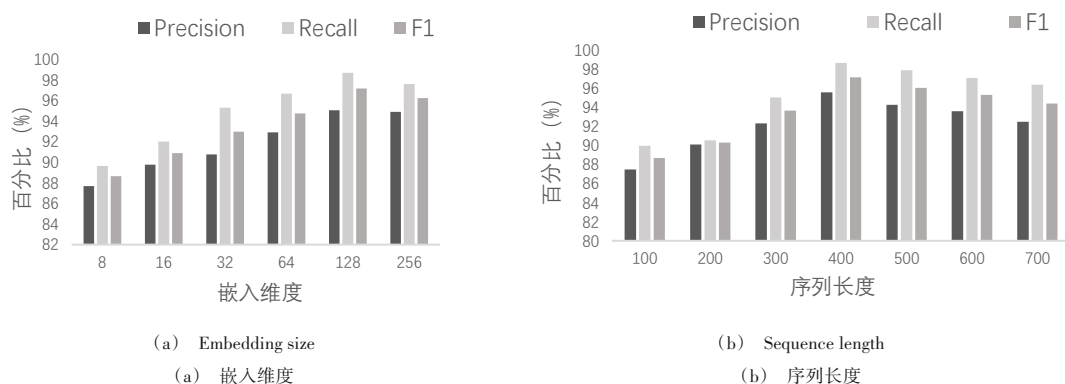


Fig. 3 Detection effect under different hyperparameters

图 3 不同参数的检测效果

3.2.4 模型效率分析

STANet 使用离线方法进行训练,为了评估其检测效率,进行了多阶段评估,包括溯源图构建、序列处理以及模型训练和测试。本文将其与 ATLAS 模型的效率进行了对比分析,结果如表 6 所示。STANet 处理单个数据集(平均大小为 676.5 MB 系统日志)以构建溯源图的平均耗时为 4 min。在训练阶段,构建并采样序列耗时约 24 min,以确保训练集中攻击序列和正常序列数量对等,而模型的学习过程耗时约 28 min,整个训练过程总耗时不超过 1 h。在测试阶段,溯源图的构建仍需 4 min,而序列构建仅需 4 s。通过遍历溯源图中的未知节点完整列表,异常判断平均仅需 1 s 即可完成。

Table 6 Average time to train and test the model

表 6 训练和测试模型的平均时间

方法	阶段	溯源图构建	序列处理	模型学习	模型检测	总时间
STANet	训练	0:04:00	0:24:12	0:28:00	-	0:56:12
	测试		0:00:04	-	0:00:01	0:04:05
ATLAS	训练	0:04:11	0:26:12	0:28:26	-	0:58:49
	测试		0:00:04	-	0:00:01	0:04:16

此外,在遇到未知攻击时,不需要重新训练模型,可以使用增量学习的方法将新攻击的特征添加到现有模型中。增量学习是一种机器学习方法,它允许模型在已有知识基础上,通过逐步接收新数据来更新自身,而无需重新训练整个模型。这种方法模仿了人类的学习模式,能够维护和增强以前的知识,同时适应新信息。这不仅节省了时间与计算资源,还能够快速适应持续变化的安全威胁环境。

4 结语

本文提出了一种基于溯源图的 APT 攻击检测方法 STANet,该方法融合了 Word2Vec、时间卷积网络和注意力机制,旨在从系统审计日志中高效检测 APT 攻击。STANet

利用 Word2Vec 和位置编码生成语义增强的特征向量,通过组合时间卷积网络和注意力机制,有效捕获溯源图中的长期因果关系特征和全局语义信息。在 10 个 APT 攻击数据集上的评估结果表明,STANet 在 APT 攻击实体检测中的精度为 95.39%,能够精确识别出攻击节点。然而,本文关注的是序列之间的动态变化,未能充分考虑溯源图的结构变化,未来工作将对 STANet 进行扩展,融合溯源图的结构演化特征,以进一步增强对复杂多变的 APT 攻击的检测能力。

参考文献:

- [1] YANG X Z, PENG G J, LIU S D, et al. Survey on attribution and inference research for APT attacks [J]. Journal of Software, 2025, 36(1): 203-252.
杨秀璋,彭国军,刘思德,等.面向 APT 攻击的溯源和推理研究综述[J].软件学报,2025,36(1):203-252.
- [2] LENG T, CAI L J, YU A M, et al. Review of threat discovery and forensic analysis based on system provenance graph[J]. Journal on Communications, 2022, 43(7): 172-188.
冷涛,蔡利君,于爱民,等.基于系统溯源图的威胁发现与取证分析综述[J].通信学报,2022,43(7):172-188.
- [3] QIU J, CHEN R R, ZHU H J, et al. A survey of network attack investigation based on provenance graph[J]. Acta Electronica Sinica, 2024, 52(7): 2529-2556.
仇晶,陈荣融,朱浩瑾,等.基于溯源图的网络攻击调查研究综述[J].电子学报,2024,52(7):2529-2556.
- [4] HOSSAIN M N, MILAJERDI S M, WANG J, et al. SLEUTH: real-time attack scenario reconstruction from COTS audit data [C]//Proceedings of the 26th USENIX Conference on Security Symposium, 2017: 487-504.
- [5] HOSSAIN M N, SHEIKHI S, SEKAR R. Combating dependence explosion in forensic analysis using alternative tag propagation semantics [C]//Proceedings of the 2020 IEEE Symposium on Security and Privacy, 2020: 1139-1155.
- [6] MILAJERDI S M, GJOMEMO R, ESHETE B, et al. HOLMES: real-time apt detection through correlation of suspicious information flows [C]//2019 IEEE Symposium on Security and Privacy, 2019: 1137-1152.
- [7] MARTIN L. Gaining the advantage: applying cyber kill chain methodology

- gy to network defense [EB/OL]. https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/Gaining_the_Advantage_Cyber_Kill_Chain.pdf.
- [8] MILAJERDI S M, ESHETE B, GJOMEMO R, et al. POIROT: aligning attack behavior with kernel audit records for cyber threat hunting [C]//Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019: 1795–1812.
- [9] LI Z, WEI Y, SHEN X, et al. MARLIN: knowledge-driven analysis of provenance graphs for efficient and robust detection of cyber attacks [DB/OL]. <https://arxiv.org/abs/2403.12541>.
- [10] HAN X Y, PASQUIER T, BATES A, et al. UNICORN: runtime provenance-based detector for advanced persistent threats [C]//27th ISOC Network and Distributed System Security Symposium, 2020: 1–15.
- [11] ALSAHEEL A, NAN Y, MA S Q, et al. ATLAS: a sequence-based learning approach for attack investigation [C]//Proceedings of the 30th USENIX Security Symposium, 2021: 3005–3022.
- [12] LIANG R Z, GAO Y, ZHAO X B. Sequence feature extraction-based APT attack detection method with provenance graphs [J]. Scientia Sinica (Informationis), 2022, 52(8): 1463–1480.
梁若舟,高跃,赵曦滨. 基于序列特征提取的溯源图上APT攻击检测方法[J]. 中国科学:信息科学, 2022, 52(8): 1463–1480.
- [13] YANG F, XU J C, XIONG C L, et al. PROGRAPHER: an anomaly detection system based on provenance graph embedding [C]//Proceedings of the 32nd USENIX Conference on Security Symposium, 2023: 4355–4372.
- [14] LI Y C, LUO H, WANG X Y, et al. Construction of advanced persistent threat attack detection model based on provenance graph and attention mechanism [J]. Journal on Communications, 2024, 45(3): 117–130.
- 李元诚, 罗昊, 王欣煜, 等. 基于溯源图和注意力机制的APT攻击检测模型构建[J]. 通信学报, 2024, 45(3): 117–130.
- [15] BHATTARAI B, HUANG H H. Prov2vec: learning provenance graph representation for anomaly detection in computer systems [C]//Proceedings of the 19th International Conference on Availability, Reliability and Security, 2024: 1–14.
- [16] WANG S, WANG Z L, ZHOU T, et al. THREATTRACE: detecting and tracing host-based threats in node level through provenance graph learning [J]. IEEE Transactions on Information Forensics and Security, 2022, 17: 3972–3987.
- [17] CHENG Z J, LV Q J, LIANG J Y, et al. Kairos: practical intrusion detection and investigation using whole-system provenance [C]//45TH IEEE Symposium on Security and Privacy, 2024: 3533–3551.
- [18] JIA Z, XIONG Y, NAN Y, et al. Magic: detecting advanced persistent threats via masked graph representation learning [C]//Proceedings of the 33rd USENIX Security Symposium, 2024: 5197–5214.
- [19] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient estimation of word representations in vector space [DB/OL]. <https://arxiv.org/abs/1301.3781>.
- [20] BAI S, KOLTER J Z, KOLTUN V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling [DB/OL]. <https://arxiv.org/abs/1803.01271>.
- [21] VASWANI A. Attention is all you need [J]. Advances in Neural Information Processing Systems, 2017, 30: 5998–6008.
- [22] BEIJERING K, GOOSKENS C, HEERINGA W. Predicting intelligibility and perceived linguistic distance by means of the levenshtein algorithm [J]. Linguistics in the Netherlands, 2008, 25(1): 13–24.

(责任编辑:陈维捷)