计 算 机 研 究 与 发 展
Journal of Computer Research and Development
https://crad.ict.ac.cn
62(10): 2495−2511, 2025

低时延智能体网络中基于三阶段关联验证的伪造链路攻击检测

张 峰 俸 皓 周秀民 章茂淳 陈镇铭 刘玉明
(桂林电子科技大学计算机与信息安全学院 广西桂林 541004)
(zhangfeng@guet.edu.cn)

Tri-Phase Correlation Verification Based Fabricated Link Attacks Detection in Low-Latency Agentic Networks

Zhang Feng, Feng Hao, Zhou Xiumin, Zhang Maochun, Chen Zhenming, and Liu Yuming
(*School of Computer Science and Information Security*, *Guilin University of Electronic Technology*, *Guilin*, *Guangxi* 541004)

Abstract The critical challenge of detecting fabricated link attacks in low-latency agentic network environments, where conventional detection methods face significant limitations is investigated in this paper. The fundamental difficulty stems from three key factors: 1) the measured performance differences between normal and fabricated links often exhibit marginal variations that fall below meaningful detection thresholds; 2) dynamic background traffic patterns introduce substantial interference that obscures genuine attack signatures; and 3) coordinated overload attacks deliberately distort measurement accuracy. These factors collectively degrade the detection performance of existing methods by significantly reducing the observable distinction between legitimate and malicious links. To address these challenges, we propose a novel three-phase detection framework based on correlated link verification. The first phase implements an effective delay transformation method that reduces measurement errors while converting raw measurements into meaningful link performance metrics. The second phase employs a multipath transmission simulation technique that amplifies subtle performance disparities between correlated links into statistically observable features. The final phase establishes robust detection thresholds through the combined application of extreme value theory and probability distribution fitting methods. Comprehensive simulation results validate that our proposed method consistently outperforms existing approaches across diverse network scales and attack scenarios. The framework demonstrates particular effectiveness in low-latency agentic environments, maintaining superior detection performance while meeting stringent timing requirements. Specifically, the framework achieves a 20% to 30% improvement in $F1$ score compared with state-of-the-art methods. Notably, even under ultra-low latency conditions (e.g., 0.1ms link delay), the proposed method maintains a high detection rate of 95.23% with a false positive rate as low as 0.28%. These advancements provide a practical solution for securing next-generation autonomous networks against sophisticated link fabrication attacks.

Key words agentic network; low-latency; fabricated link attacks; correlated link verification; extreme value theory

摘 要 面向低时延的智能体网络环境,研究了一种基于关联链路验证的伪造链路攻击检测方法.在该网络环境下,正常链路与伪造链路在测量时延等关键指标上的统计特征差异较小,并且背景流量和协调器过载攻击的影响会放大测量误差,但缩小该差异会导致现有方法的检测性能骤降.对此,提出一种基于关

收稿日期: 2025-06-04; **修回日期**: 2025-07-16
基金项目: 广西重点研发计划项目(AB25069302); 南宁市邕江计划项目(2023018)
This work was supported by the Key Research and Development Program of Guangxi (AB25069302) and the Nanning Yongjiang Program (2023018).
通信作者: 刘玉明(ymliu@guet.edu.cn)

联链路验证的伪造链路攻击检测方法,该方法包含3个阶段:首先,采用一种有效时延转换方法缓解测量误差的影响,并将处理后的测量时延等转化为链路性能指标;然后,根据提出的多径传输模拟方法,将关联链路之间的性能差异转化为易于观测的统计特征;最后,基于极值理论和概率分布拟合方法确定统计特征的阈值,用于伪造链路的检测.仿真实验结果表明,在不同的网络规模和攻击场景下,所提出的方法能够有效地检测低时延智能体网络中的伪造链路攻击,并且在检测性能方面明显优于现有的相关方法,*F*1分数提升达20%~30%.特别地,当链路传输时延低至0.1 ms时,仍然能取得95.23%的检测率和0.28%的误判率,显示出良好的鲁棒性.

关键词 智能体网络;低时延;伪造链路攻击;关联链路验证;极值理论

中图法分类号 TP311.13; TP309

DOI: 10.7544/issn1000-1239.202550399 **CSTR:** 32373.14.issn1000-1239.202550399

随着多智能体系统(multi-agent systems, MAS)与人工智能服务的持续融合,智能体网络(agentic network)逐渐发展成为一种面向智能体自治协同的新型网络范式,并已在智能工厂、智慧城市等多种AIoT场景中得到广泛应用[1-2].该架构旨在支持异构智能体在跨域环境中实现高效的协同感知、推理决策与服务执行,具有高度自主性和去中心化的特征[3].在实际部署中,智能体网络通常通过统一的注册机制与发现协议,将大量分布式智能体接入网络,并为其分配唯一的数字身份.这一过程多基于应用层协议,如agent network protocol(ANP),完成智能体的注册、认证、能力通告及调度协作等功能[4].

尽管上述逻辑层机制在一定程度上提升了智能体间交互的可信性与协同性,智能体网络的安全性仍面临多维威胁.根据攻击行为的作用层级,可将其大致划分为2类:一类是链路层攻击,主要针对网络底层的物理与逻辑连接关系,典型代表如伪造链路、链路劫持等;另一类则是应用层攻击,例如恶意智能体注册、能力伪报、任务拒绝响应等行为,直接破坏调度协作的信任基础.

目前已有的安全研究多集中于应用层,重点关注智能体行为验证、调度策略鲁棒性和恶意智能体识别等方向[5],而对链路层安全问题的研究相对不足.当前应用层的安全主要通过身份管理、加密、认证和授权等机制进行保证,交互过程通常以秒为单位,考虑到智能体网络中智能体的动态特性(可能频繁地加入、离开网络),在以智能工厂为代表的需要以毫秒为单位的典型低延时应用场景中,这些方法将不再适用.本文正是聚焦于这一被忽视的层面,研究其潜在风险与防御方法.由于智能体网络在底层通信中仍依赖传统网络设施的链路结构,尤其在智能体初次接入或协作路径更新过程中,控制面需通过链路发现机制掌握当前的物理拓扑.这一过程极易遭受利用拓扑发现协议设计缺陷发起的伪造链路攻击.以软件定义网络(software-defined networking, SDN)中的链路层发现协议(如LLDP)为例,已有研究表明攻击者可以通过中继或伪造探测报文,诱使控制器误判2个物理上并未连接的节点之间存在链路[6].类似方式在智能体网络中同样适用,可能导致协调器构建出错误的逻辑拓扑视图,进而引发一系列严重后果[7-8].一方面,路径选择与资源调度将在不可信拓扑信息基础上执行,导致通信延迟增加和任务分配失衡;另一方面,推理数据路径被劫持或重构,可能破坏感知输入与语义一致性,降低系统推理精度与稳定性.更为严重的是,攻击者可借助伪造链路将受控智能体引入协同推理过程,操纵中间结果、误导模型更新,甚至植入后门知识,从而将智能体网络由自治协作系统演变为可被远程操控的攻击工具,对关键基础设施构成长期隐蔽威胁.

这一问题在面向智能制造、车联网等对低时延与可靠性要求极高的AIoT场景中更为突出.在这类场景中,智能体之间的协同感知、任务分派与执行反馈高度依赖于底层链路的时延与稳定性.因此,链路质量的准确感知成为控制面进行调度与路径规划的关键基础.一旦攻击者利用控制器在处理探测报文时产生的“抖动”误差,构造出具有真实延迟特征的伪造路径,便可有效绕过基于延迟分布建模或最短路径一致性分析的传统检测机制[9-10].此外,由于伪造链路攻击不会显著增加网络流量或引起通信负载异常,现有依赖流量特征或行为模式的检测方法难以奏效[11-12].

为应对上述挑战,本文提出了一种面向智能体网络的链路层攻击检测方法——CLV(correlated link verification).该方法基于“关联链路”概念,评估由同

一智能体发起、处于相似网络条件下的多个邻接链路在通信时延特性上的一致性，从而识别出存在异常偏差的可疑伪造路径. CLV 方法具备部署轻量、信息依赖低等优势，即无需引入额外控制消息，不依赖高维行为建模，仅通过简单的通信延迟测量即可实现高效且可扩展的拓扑安全检测.

1 相关工作

针对伪造链路的检测，目前主流的检测方法为静态或动态的链路时延检测方法，它们的主要区别在于是否更新检测阈值. 静态方法需在系统运行前进行部署，以采集全网或部分链路的时延作为基线数据集，并以此构建基于统计或机器学习的判定模型. 在系统运行后，根据采集的链路数据判定新增链路是否是伪造的. 动态方法在系统运行中即可部署，通过实时采集的链路时延更新检测阈值，以对新增的链路的合法性进行判定. 另外，还有一些检测方法需要对攻击场景进行特定的假设，因此具有一定的局限性.

在静态的检测方法中，Smyth 等人[13]较早地关注了 SDN 环境下的伪造链路攻击，通过分析不同中继节点数量以及中继模块处理能力情况下伪造链路时延与正常链路时延分布的差异，提出了基于 p 值的检测方法. 为增强检测方法的可扩展性，Al-Salti 等人[14]设计了 LINK-GUARD，通过采集不同网络规模下的链路时延来构建数据集，并基于箱线图方法检测异常链路. 然而，由于未考虑网络波动与控制器扰动的影响，这些方法缺乏鲁棒性. 为此，Soltani 等人[15]设计了 MLLG，它不仅考虑了交换机之间的链路时延，也考虑了交换机与控制器之间的链路时延. 利用这些时延构建数据集，并采用机器学习的方法训练判别模型. 进一步地，Soltani 等人[16]设计了 RLV，通过采集不同网络规模、不同伪造链路数量和是否包含引入背景流量等情景下的链路时延，以训练更具扩展性和鲁棒性的判别模型. 然而，在低时延的网络环境下，网络波动与控制器扰动对测量结果的干扰尤为明显，这些方法的检测效果仍然不够理想，难以有效地发现伪造链路攻击.

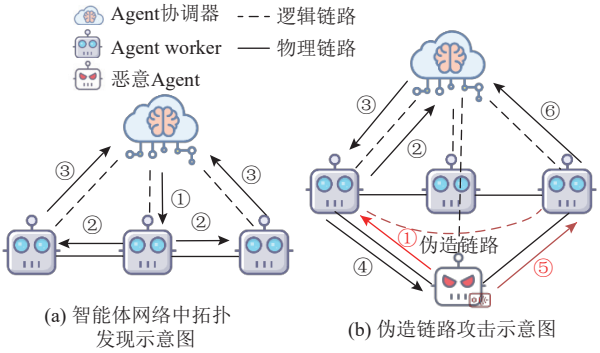
在动态的检测方法中，Skowyra 等人[17]设计了 TopoGuard+，使用全局队列存储采集到的链路时延，并根据定义的阈值计算方法动态地更新检测阈值. 若链路的测量时延超过阈值，则认为该链路是伪造的，随后启用防御模块短暂地停用该链路. 但如文献 [18] 所述，TopoGuard+中的检测方法存在缺陷. 攻击者可通过扰动控制器来增大测量误差和检测阈值，进而绕过检测. Chou 等人[19]和 Huang 等人[20]分别设计了 CTAD 和 TrustTopo，通过修正阈值的计算方法，来降低阈值的变化幅度，在一定程度上提高了检测率. 然而，上述动态检测的方法均难以设计合适的阈值，以准确地区分正常链路和伪造链路，仍然缺乏鲁棒性.

此外，还有一些基于特定假设的检测方法. Shrivastava 等人[21]设计了 SRD，在假设中继模块无法处理巨帧的前提下，利用巨型探测报文探测发现伪造链路. Wang 等人[22]假设伪造链路攻击会引起大量会话中断，并提出基于强化学习的检测方法. Kong 等人[23]提出一种具有攻击维持能力的伪造链路攻击模型，能够突破现有方法的检测，并对此提出一种基于转发路径验证的检测方法. 这些方法对攻击者的能力进行了明确限制，缺乏普适性. 此外，由于伪造链路攻击本身不破坏传输报文的完整性，一些基于安全协议[24-28]的拓扑欺骗防御方法，以及针对主机位置欺骗攻击的防御方法[29-31]将无法适用.

2 问题描述与建模

2.1 问题描述

智能体网络中探测数据包的转发过程如图 1(a)所示. 首先，智能体协调器构造探测数据包，将其发送至各 Agent worker(步骤①). 接着，Agent worker 收到探测数据包后，通过广播的形式将其转发至与之连接的其他 Agent worker(步骤②). 当某个 Agent worker 收到来自其邻居的探测数据包后，将其转发至 Agent 协调器(步骤③). 最后，Agent 协调器根据闭环的探测数据包转发路径，确定 Agent worker 之间的连接，



(a) 智能体网络中拓扑发现示意图 (b) 伪造链路攻击示意图

Fig. 1 Illustration of topology discovery and fabricated link attacks in agentic networks

图 1 智能体网络中拓扑发现与伪造链路攻击示意图

进而构建全局的拓扑视图.

然而, 上述链路发现过程可能受到伪造链路攻击的威胁. 由于探测数据包广播传递, 恶意 Agent worker 能够在正常 Agent worker 之间中继该消息, 使 Agent 协调器误以为正常 Agent worker 之间存在一条链路, 该链路实际由恶意 Agent worker 伪造. 具体流程如图 1(b)中步骤③~⑥所示. 基于该伪造链路, 恶意 Agent worker 可能丢弃、监听、篡改正常 Agent worker 之间的业务通信数据.

尽管有研究工作通过探测数据包测量链路时延, 并根据异常的链路时延值发现伪造链路, 但它们难以适用于低时延的网络环境. 在低时延场景下, Agent 协调器测量的链路时延容易受到背景流量等因素的影响. 不仅如此, 恶意 Agent worker 可能向 Agent 协调器发起过载攻击, 发送大量查询请求增加其负载, 影响链路时延的测量结果, 如图 1(b)中的步骤①②所示. 在此背景下, 链路时延测量误差被放大, 导致现有方法检测性能急剧下降, 难以准确检测伪造链路的存在.

2.2 威胁建模

本节首先建立测量的链路时延的模型, 分析伪造链路攻击的特点以及检测该攻击存在的挑战. 接着, 给出现有方法的形式化描述, 分析它们在低时延网络中难以准确识别伪造链路的原因. 本节主要使用的符号及其含义如表 1 所示.

Table 1 Notations and Description

表 1 符号及描述

符号	含义
T	关于时延的随机变量
D	用于训练模型的数据集
I	示性函数
Z	决策函数, 判断链路是否为伪造链路
δ	检测阈值
d	链路时延
l	网络中的一条链路
w	链路质量的评分, 取值为 0~1
S	由 0 和 1 组成的轨迹序列, 表示多径传输模拟方法的选路结果
H	轨迹序列中 0 和 1 分布的香农熵值
K	轨迹序列中连续 1 的最大长度

根据图 1 可知, 正常链路的测量时延 T_{nor} 和伪造链路的测量时延 T_{mal} 分别如式(1)和式(2)所示.

$$T_{\text{nor}} = T_{\text{trx}} + T_{\text{fwd}}^{\text{s}} + T_{\text{err}}, \tag{1}$$

$$T_{\text{mal}} = (m+1)T_{\text{trx}} + \sum_{i=1}^{m} T_{\text{fwd}}^{\text{h},i} + T_{\text{err}}. \tag{2}$$

其中, T_{trx} 表示链路传输时延, $T_{\text{fwd}}^{\text{s}}$ 和 $T_{\text{fwd}}^{\text{h}}$ 分别表示 Agent worker 端口和中继模块的转发时延, T_{err} 表示测量误差, m 表示中继模块的数量. 由于传播时延非常小, 故此处将其忽略. 为便于分析, 假设 T_{trx} 为常量 c, T_{err} 的分布未知. 假设报文按速率为 λ_{lnk} 的泊松过程到达, Agent worker 端口和中继模块分别按速率为 $\lambda_{\text{fwd}}^{\text{s}}$ 和 $\lambda_{\text{fwd}}^{\text{h}}$ 的泊松过程转发. 通常情况下, 中继模块相较于 Agent worker 端口的转发速率更低, 在链路不过载的情况下有 $\lambda_{\text{lnk}} < \lambda_{\text{fwd}}^{\text{h}} < \lambda_{\text{fwd}}^{\text{s}}$, 此时 Agent worker 端口和中继模块的平均转发时延分别为 $1/(\lambda_{\text{fwd}}^{\text{s}} - \lambda_{\text{lnk}})$ 和 $1/(\lambda_{\text{fwd}}^{\text{h}} - \lambda_{\text{lnk}})$. 因此, 正常链路和伪造链路的测量时延的期望分别如式(3)和式(4)所示.

$$E(T_{\text{nor}}) = c + \frac{1}{\lambda_{\text{fwd}}^{\text{s}} - \lambda_{\text{lnk}}} + E(T_{\text{err}}), \tag{3}$$

$$E(T_{\text{mal}}) = (m+1)c + \sum_{i=1}^{m} \frac{1}{\lambda_{\text{fwd}}^{\text{h}} - \lambda_{\text{lnk}}} + \frac{1}{\lambda_{\text{fwd}}^{\text{s}} - \lambda_{\text{lnk}}} + E(T_{\text{err}}). \tag{4}$$

通过对比式(3)(4)可知, $\Delta = E(T_{\text{mal}}) - E(T_{\text{nor}}) > 0$, 这表明正常链路与伪造链路之间存在固有的性能差异, 这种差异性正是相关方法检测伪造链路的基础. 当满足如下 3 种情况时, 差异 Δ 是较为显著的: 1)传输时延较大, 即 c 取值较大时的情况. 2)中继模块的性能较低, 即 $\lambda_{\text{fwd}}^{\text{h}}$ 取值较小时的情况. 此时, 中继模块的转发时延较高. 3)伪造链路过载, 即 $\lambda_{\text{fwd}}^{\text{h}} < \lambda_{\text{lnk}}$ 时的情况. 此时式(4)不再成立, 并且转发时延会显著增加. 然而, 在低时延网络中, 攻击场景可能不再满足上述 3 种情况. 相应地, 现有的检测方法难以准确地识别伪造链路. 根据阈值是否动态更新, 现有方法可分为基于静态阈值和动态阈值的检测方法, 其形式化描述和脆弱性分析如下.

静态检测方法的思想在于, 通过采集的数据集训练检测模型, 以识别伪造链路, 其形式化描述如式(5)所示.

$$Z_{\text{sta}}(l_i) = I\left(\sum_{t=1} g_{\text{tnt}}\left(D_{\text{sta}}, \boldsymbol{v}_{l_i}^{t}\right) \geqslant \delta_{\text{n}}\right), \tag{5}$$

其中, $D_{\text{sta}} = \{\{\boldsymbol{v}_{l_1}^{1}, \boldsymbol{v}_{l_2}^{1}, \cdots\}, \{\boldsymbol{v}_{l_1}^{2}, \boldsymbol{v}_{l_2}^{2}, \cdots\}, \cdots\}$ 表示在检测模型部署前采集的数据集, 用于模型的训练. $\boldsymbol{v}_{l_i}^{t}$ 表示链路 l_i 在第 t 个采样周期的特征向量, 用于模型的测试. g_{tnt} 表示训练和测试函数, 它根据 D_{sta} 和 $\boldsymbol{v}_{l_i}^{t}$ 得到一次判别结果. Z_{sta} 表示决策函数, 根据累计的判别结果与阈值 δ_{n} 的关系得到最终的检测结果. 不同的静态检测方法

所使用的数据集、检测模型以及决策阈值各不相同. 在文献 [13] 中，$\boldsymbol{v}_{l_i}^t=(d_{l_1}^t,d_{l_2}^t,\cdots,d_{l_i}^t)$，$g_{\text{tnt}}=p-value$，$\delta_{\text{n}}>1$，其中$d_{l_i}^t$表示在第$t$个采样周期链路$l_i$的测量时延. 在文献 [15] 中，$\boldsymbol{v}_{l_i}^t=(d_{l_i}^t,d_{\text{cs}}^t,d_{\text{sc}}^t)$，$g_{\text{tnt}}=ML$，$\delta_{\text{n}}=1$，其中$d_{\text{cs}}^t$和$d_{\text{sc}}^t$分别表示控制器到交换机以及交换机到控制器的测量时延，ML表示机器学习的模型. 在文献 [14] 中，$\boldsymbol{v}_{l_i}^t=(d_{l_1}^t,d_{l_2}^t,\cdots,d_{l_i}^t)$，$g_{\text{tnt}}=Q3+3\times IQR$，$\delta_{\text{n}}=10$，其中$Q3$和$IQR$分别表示测量时延集合的第 3 个四分位数和四分位数间距.

动态检测方法的思想在于，通过采集的数据集和动态的时延阈值来识别伪造链路，其统一的形式化描述如式(6)所示.

$$Z_{\text{dyn}}(l_i)=I\left(\sum_{t=1}I\left(d_{l_i}^t>\delta_{\text{d}}^t\right)\geqslant\delta_{\text{n}}\right),\tag{6}$$

其中，$d_{l_i}^t$表示在第t个采样周期链路l_i的测量时延，δ_{d}^t表示动态的时延阈值. Z_{dyn}表示决策函数，它根据测量时延$d_{l_i}^t$和时延阈值δ_{d}^t之间的大小关系得到一次判别结果，并根据累计的判别结果与决策阈值δ_{n}得到最终的检测结果. 时延阈值δ_{d}^t通过$\delta_{\text{d}}^t=g_{\text{thr}}(D_{\text{dyn}})$进行计算. 其中，$g_{\text{thr}}$为时延阈值的计算函数，$D_{\text{dyn}}=\{\{d_{l_1}^1,d_{l_2}^1,\cdots\},\cdots\{d_{l_1}^t,d_{l_2}^t,\cdots\}\}$表示截至第$t$个采样周期采集到的数据集. 不同的动态检测方法使用的阈值计算函数和决策阈值各不相同. 在文献 [17] 中，$g_{\text{thr}}=Q3+3\times IQR,\delta_{\text{n}}=1$. 在文献 [19] 中，$g_{\text{thr}}=Q3+1.5\times IQR,\delta_{\text{n}}=1$. 在文献 [20] 中，$g_{\text{thr}}=\bar{d}_{l_i}+\alpha\max(d_{l_i}^t-\bar{d}_{l_i},0)+c,\delta_{\text{n}}\geqslant1$，其中$\bar{d}_{l_i}$表示平均时延，$\alpha$表示测量结果超过均值$\bar{d}_{l_i}$的比例，$c$为修正阈值的常数项.

最后，分析这些检测方法的脆弱性. 由式(5)可知，静态检测方法的检测性能主要取决于数据集D_{sta}的质量. 然而，在低时延网络中测量误差是不容忽视的，它可能污染数据集和削弱模型的检测效果. 另外，静态检测方法适应性较差，针对一个环境的数据集和模型难以直接应用于其他环境，相比之下动态检测方法适应性更强. 由式(6)可知，动态检测方法的检测性能取决于动态阈值δ_{d}^t的选取. 然而，现有方法直接根据测量结果的数值大小或统计特征来确定阈值缺乏鲁棒性. 攻击者可通过发起过载攻击增大测量误差，进而增大阈值δ_{d}^t，使得$d_{l_i}^t<\delta_{\text{d}}^t$来规避检测[15-16,18]. 不仅如此，攻击者可通过优化中继模块的性能来降低转发时延，缩小伪造链路与正常链路之间的性能差异. 此时，确定合适的δ_{d}^t是有挑战的，δ_{d}^t过高会导致较低的检测率，而过低则会导致较高的误报率.

3 基于关联链路验证的伪造链路检测方案

3.1 系统模型

如图 2 所示，关联链路指从同一 Agent worker 出行的 2 条处理探测数据包的链路，考虑到广播的探测数据包几乎在相同时刻沿关联链路发出，正常情况下 2 条链路的测量结果差异是相对确定且较小的. 基于该思路，CLV 检测方法的工作流程如图 3 所示. 需要说明的是，图 2 中展示的网络拓扑仅为便于阐述关联链路的概念而进行的简化表示，并不意味着各 Agent worker 必须仅连接 2 个邻居. 实际上，其与任意 2 个邻居节点之间的出行链路均可构成一组关联链路.

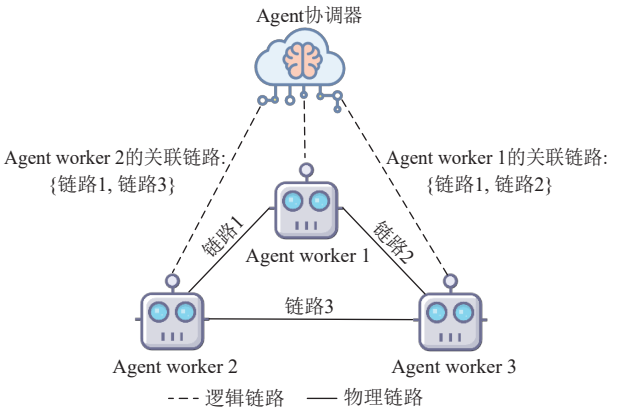


Fig. 2 Illustration of correlated links

图 2 关联链路示意图

在攻击检测之前，通过提出的数据处理方法，将采集的原始数据经过处理后用来转化链路质量，目的是缓解测量误差，以准确地反映实际的链路性能，详见 3.2 节的内容. 在攻击检测阶段，CLV 首先通过提出的多径传输模拟方法，将测量的关联链路之间的性能差异转化为易于观测的统计特征，详见 3.3 节的内容. 接着，根据历史的统计特征序列，基于极值理论和概率分布拟合的方法确定统计特征的阈值，详见 3.4 节的内容. 最后，获取关联链路当前时刻的统计特征，并结合确定的阈值识别其中的伪造链路.

3.2 数据预处理

为评估链路的实时性能，引入链路质量的概念. 根据式(4)可知，链路时延和可承受负载可作为链路质量的评估指标，且低时延与高负载均能指示更优的链路质量. 需要说明的是，在低时延网络中，高时延表明测量误差是显著的，故仅当测量的链路时延较低时，链路负载才能有效地指示链路质量的优劣.

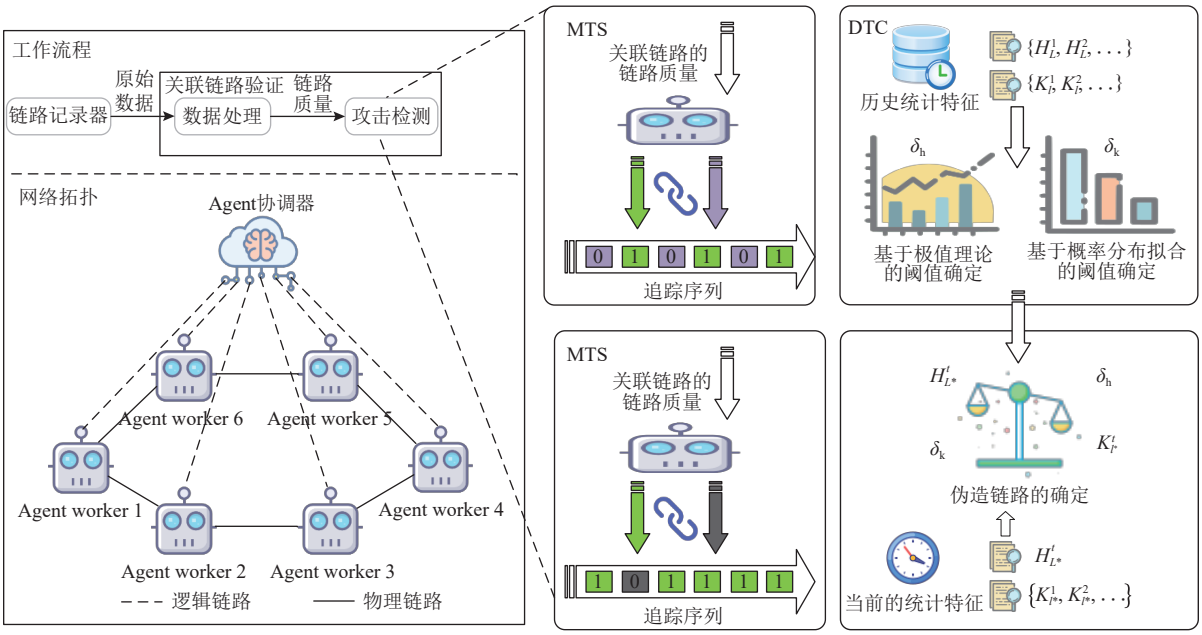


Fig. 3 Ilustration of fabricated link detection method based on correlated link verification

图 3 基于 CLV 的伪造链路检测方法示意图

因此, 给出链路质量的定义如式(7)所示. 表示在第t个采样周期链路的时延评分如式(8)所示. 表示链路的负载评分如式(9)所示.

$$w_l^t=\frac{w_{\mathrm{d},l}^t\left(1+\exp\left(w_{\mathrm{d},l}^t-1\right)\times w_{\lambda,l}^t\right)}{2},\tag{7}$$

$$w_{\mathrm{d},l}^t=1-\tanh\left(\frac{\widetilde{d}_l^t}{d_{\mathrm{lim}}}\right),\tag{8}$$

$$w_{\lambda,l}^t=\frac{1}{1+\exp\left(\alpha\lambda_{\max}-\widetilde{\lambda}_l^t\right)}.\tag{9}$$

在式(8)中, d_l^t表示在第t个采样周期链路l的测量时延, d_{lim}表示有效时延的上界, $\widetilde{d}_l^t$表示经过转化的有效时延. 其意义在于, d_{lim}用于指示测量时延的有效性, 即仅当测量时延低于该值时被认为有效, 否则该测量时延被认为不能有效反映实际链路性能. $\widetilde{d}_l^t$根据算法 1 和输入d_l^t获得. 具体地, 若当前测量时延有效, 则通过移动平均的方法得到最近几个采样周期的平均时延, 如行⑨~⑪所示. 反之, 继续判断有效时延队列的长度是否达到统计要求, 若达到则取有效时延队列的中值, 否则取真实时延队列的中值, 如行③~⑧所示.

算法 1 借鉴了缺失值的处理思想, 将超过d_{lim}的测量时延值视作缺失值. 由于时延评分对于低时延值更加敏感, 故采用tanh函数归一化时延评分. 在式(9)中, λ_l^t表示在第t个采样周期链路l的测量负载. $\widetilde{\lambda}_l^t$表示经过转化的有效负载, 转化的方法与算法 1 一致. $\lambda_{\max}$表示链路带宽, 当$0<\alpha\leqslant 1$时, $\alpha\lambda_{\max}$表示链路是否过载的界限. 由于当链路负载超过$\alpha\lambda_{\max}$, 转发时延会陡然增加, 负载评分需相应地快速降低, 因此采用sigmoid函数归一化负载评分.

算法 1. 有效时延转化算法.

输入: 测量的链路时延d_l^t, 有效时延的限值d_{lim};

输出: 经过转化的有效链路时延$\widetilde{d}_l^t$.

① 若系统处于初始状态, 则初始化有效时延队列*valid_latency_queue*和真实时延*real_latency_queue*为空;

② 将d_l^t添加至循环队列*real_latency_queue*;

③ if $d_l^t>d_{\mathrm{lim}}$ then

④ if *valid_latency_queue*长度不够 then

⑤ $\widetilde{d}_l^t\leftarrow median\left(real_latency_queue\right)$;

⑥ else

⑦ $\widetilde{d}_l^t\leftarrow median\left(valid_latency_queue\right)$;

⑧ end if

⑨ else

⑩ 将d_l^t添加至循环队列*valid_latency_queue*;

⑪ $\widetilde{d}_l^t\leftarrow moving_average\left(valid_latency_queue\right)$;

⑫ end if

3.3 基于多径传输模拟的统计特征构建方法

为便于用统计特征刻画关联链路之间的性能差异, 考虑将关联链路上每个周期的采样结果转化为离散的序列, 本节给出一种多径传输模拟(multipath transmission simulation, MTS)方法, 如算法 2 所示. 其原理如下: 引入一个虚拟的会话, 根据关联链路上每

个周期的采样结果，选择其中性能更优的一条链路传输一个分组. 经过多个采样周期后，虚拟会话的选择结果能够反映链路之间的性能差异. 同时，虚拟会话的选择结果可以通过 0 和 1 组成的轨迹序列表达，其中 1 和 0 分别表示是否选择该链路. 因此，通过分析轨迹序列的统计特征，即可评估关联链路之间的性能差异.

算法 2. 多径传输模拟算法.

输入：关联链路 $corr_links$，链路质量 w_l^t，模拟步长 $steps$，折扣率 γ_0，统计周期 T_{sta}，检测周期 T_{det}；

输出：统计特征 K 和 H 的序列 K_{seq} 和 H_{seq}.

① 初始化列表 Tr_{seq}，K_{seq}，H_{seq} 为空；

② for t from 0 to $steps$ do

③ for $l \in corr_links$ do

④ if $w_l^t \times \gamma_l^t$ 更大 then

⑤ 将 $l.id$ 添加至轨迹序列 Tr_{seq}；/*从 1 开始的 0-1 序列*/

⑥ $\gamma_l^{t+1} \leftarrow \gamma_l^t \times \gamma_0$；

⑦ else

⑧ $\gamma_l^{t+1} \leftarrow 1$；

⑨ end if

⑩ end for

⑪ end for

⑫ 分别按照统计周期 T_{sta} 和检测周期 T_{det}，将 Tr_{seq} 划分为多个子序列的集合 $Tr_{\text{seq}}^{\text{k}}$ 和 $Tr_{\text{seq}}^{\text{h}}$；

⑬ for $seq \in Tr_{\text{seq}}^{\text{k}}$ do

⑭ 将子序列 seq 中连续 1 的数量 K 添加至 K_{seq}；

⑮ end for

⑯ for $seq \in Tr_{\text{seq}}^{\text{h}}$

⑰ 将子序列 seq 中 0 和 1 的分布的熵值 H 添加至 H_{seq}；

⑱ end for

在算法 2 中，轨迹序列的获取过程如行④⑤所示. 需要说明的是，正常链路之间仍存在的十分细微的性能差异，若虚拟会话直接根据链路质量选择链路，则可能始终选择其中一条性能略优的链路，导致选择结果难以反映链路之间的性能差异. 为屏蔽链路之间这种细微的性能差异，算法 2 引入折扣因子来抑制虚拟会话连续选择同一条链路的行为. 具体地，虚拟会话根据链路质量与相应折扣因子的乘积来选择链路，当连续选择同一条链路时，折扣因子不断递减. 折扣因子的定义如式(10)所示，其更新过程如行⑥~⑧所示. 其中，γ_l^t 表示在第 t 个采样周期链路的折扣因子，γ_0 表示折扣率，b_l^t 表示在第 t 个采样周期链路是否被虚拟会话所选择.

$$\gamma_l^t = \begin{cases} 1, & b_l^t b_l^{t-1} = 0, \\ \gamma_0 \gamma_l^{t-1}, & b_l^t b_l^{t-1} = 1, \end{cases} \tag{10}$$

$$T_{\text{sta}} = T_{\text{spl}} \times \sum_{l \in L} \max_{K} \prod_{k=1}^{K} b_l^k = 1. \tag{11}$$

统计特征 K 和 H 的计算说明如行⑭和行⑰所示，其含义解释如下：记关联链路 $L = \{l_i, l_j\}$ 的轨迹序列为 $S_{l_i} = \left[b_{l_i}^0, b_{l_i}^1, \cdots\right]$，根据先前的分析，正常关联链路之间的性能差异十分细微. 因此，S_{l_i} 中 0 和 1 的数量应非常接近. 另外，折扣因子的引入旨在消除链路之间细微的性能差异，从而限制虚拟会话连续地选择同一条链路，故 S_{l_i} 中连续 1 的长度较短，如 $S_{l_i} = [1,0,1,0,1,1,0]$. 然而，伪造链路的存在使得虚拟会话倾向于选择性能更优的正常链路. 相应地，正常链路的轨迹序列中 1 的数量较多且连续 1 的长度较长，如 $S_{l_i} = [1,1,1,1,1,0,1]$. 因此，利用香农熵 H 评估轨迹序列中 0 和 1 数量的均衡程度，而令 K 表示轨迹序列中连续 1 的长度. 另外，统计特征 H 和 K 被添加至列表中，用于后续阈值确定和伪造链路检测.

在算法 2 行⑫中，轨迹序列按照统计周期和检测周期被分为多个子序列，以便于统计特征的计算. 为了获取一个统计周期里连续 1 的数量，定义统计周期 T_{sta} 如式(11)所示，其中 T_{spl} 表示采样周期. 可见，统计周期 T_{sta} 的大小是动态的，它的取值取决于统计特征 K 的大小. 另外，为了获得稳定的结果，计算多个统计周期内的熵值，即 $T_{\text{det}} = Interval \times T_{\text{sta}}$，此处 T_{det} 表示计算熵值的周期.

3.4 基于极值理论与概率分布拟合的阈值计算方法

为能够确定上述统计特征的阈值，提出一种动态阈值计算(dynamic threshold calculation, DTC)方法. 根据 3.3 节的分析可知，动态阈值应满足 2 个条件：1)当利用统计特征识别伪造链路时，阈值应使得式(12)成立. 2)当利用统计特征识别伪造链路时，阈值应使得式(13)成立. 需要说明的是，阈值的确定过程即为 CLV 方法的训练过程，而式(12)(13)中示性函数的结果即为 CLV 方法的检测结果，它们均能够指示伪造链路的存在.

$$I_{\text{h}} = \begin{cases} 1, & H \leqslant \delta_{\text{h}}, \\ 0, & H > \delta_{\text{h}}. \end{cases} \tag{12}$$

$$I_{\text{k}} = \begin{cases} 1, & K > \delta_{\text{k}}, \\ 0, & K \leqslant \delta_{\text{k}}. \end{cases} \tag{13}$$

首先，确定阈值 δ_{h}. 记熵值随机变量构成的序列

为$[H_L^1, H_L^2, \cdots]$, 假设这些随机变量相互独立且服从相同分布, 分布函数用F_{h}表示. 由于该分布未知, 故针对该序列研究其阈值u的超出量的分布, 如式(14)所示, 其中y为超出量. 需要注意的是, 阈值u是极值理论[32]的一个研究对象, 它不同于待求解的阈值δ_{h}. 由于原始的分布函数F_{h}未知, 假设阈值u服从广义的Pareto分布, 分布函数F_u如式(15)所示. 其中, ξ为形状参数, μ为位置参数(实际取$\mu=0$), $\tilde{\sigma}=\sigma+\xi(u-\mu)$为尺度参数. 文献[33]给出了参数$\hat{\xi}$和$\hat{\sigma}$的估计方法, 进而可根据式(16)计算出决策值$z_{q_{\mathrm{h}}}$. 其中, u的值根据经验设置为0.98倍样本最大值, N^*为观测样本的总数量, N_u为超过观测值超过阈值u的样本数量. 对于一个给定的危险系数q_{h}, 如$q_{\mathrm{h}}=1E-5$, 令$\delta_{\mathrm{h}}=z_{q_{\mathrm{h}}}$即可得到相应的阈值. 作为熵值序列的极值, δ_{h}满足式(12).

$$Pr\{H_L<u+y|H_L>u\}=\frac{F_{\mathrm{h}}(u+y)-F_{\mathrm{h}}(u)}{1-F_{\mathrm{h}}(u)},\quad y>0, \tag{14}$$

$$F_u=1-\left(1+\frac{\xi y}{\tilde{\sigma}}\right)^{-\frac{1}{\xi}},\quad y>0,\quad 1+\frac{\xi y}{\tilde{\sigma}}>0, \tag{15}$$

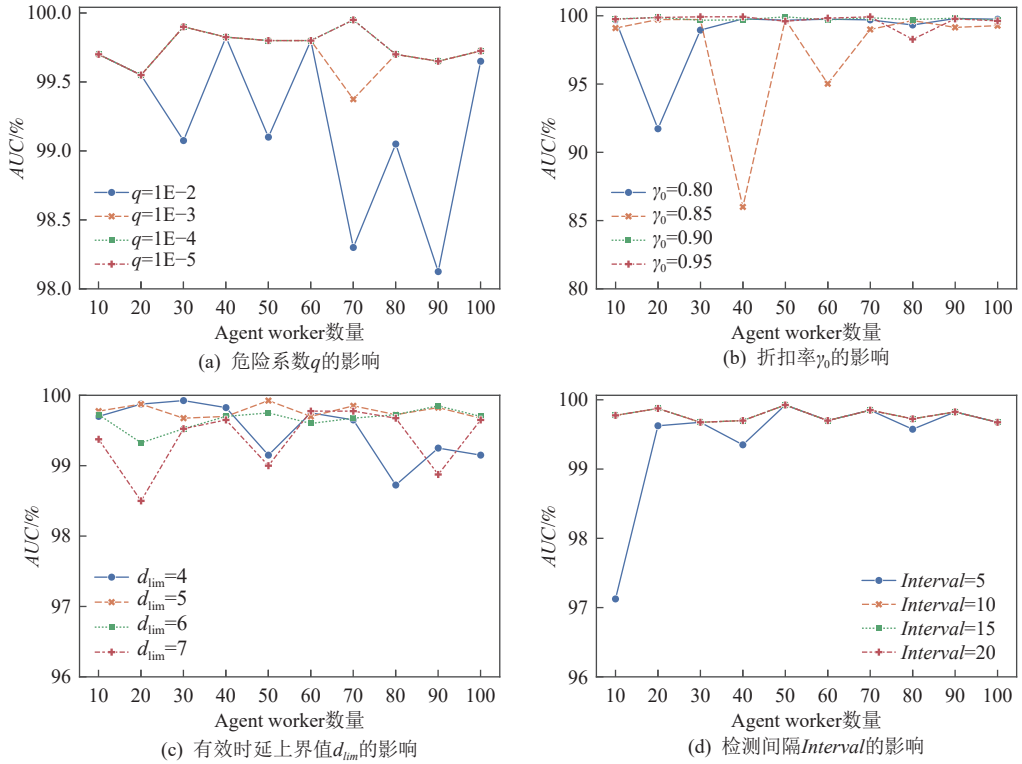
$$z_{q_{\mathrm{h}}}=u+\frac{\hat{\sigma}}{\hat{\xi}}\left(\left(\frac{N^*(1-q_{\mathrm{h}})}{N_u}\right)^{-\hat{\xi}}-1\right). \tag{16}$$

接着, 确定阈值δ_{k}. 记K值随机变量构成的序列为$[K_l^1, K_l^2, \cdots]$, 假设这些随机变量相互独立且服从相同分布, 分布函数用F_{k}表示. 由式(11)可知, 在一个统计周期内, 轨迹序列中连续1的数量为首次出现0之前1出现的次数. 这类似于抛硬币实验中, 硬币首次出现反面时正面已经出现的次数. 因此, 利用几何分布拟合随机变量K的分布, 如式(17)所示. 令p_l表示虚拟会话选择链路l的概率, 基于采集的数据集, 期望$E(K_l)$可以通过统计分析得到. 进一步, 根据一阶矩估计$E(K_l)=1/\hat{p}_l$, 可计算得到$\hat{p}_l$和$\hat{F}_{\mathrm{k}}$, 进而可计算出决策值$z_{q_{\mathrm{k}}}$, 如式(18)所示. 对于一个给定的危险系数q_{k}, 如$q_{\mathrm{k}}=1E-5$, 令$\delta_{\mathrm{k}}=z_{q_{\mathrm{k}}}+1$即可得相应的阈值. 作为分布函数$F_{\mathrm{k}}$的边界, δ_{k}满足式(13).

$$\hat{F}_{\mathrm{k}}=\sum_{i\in\mathbb{N}}(1-\hat{p})^{i-1}\times\hat{p}, \tag{17}$$

$$z_{q_{\mathrm{k}}}=\min_{\hat{F}_{\mathrm{k}}\geqslant 1-q_{\mathrm{k}}} K. \tag{18}$$

需要说明的是, 式(16)与式(18)中阈值设定依赖于参数u或q, 但二者对模型性能的影响均较为有限. 具体地, u表示尾部数据的截断阈值, 尽管其取值会影响参数σ与ξ的估计, 但在典型取值范围内(如样本最大值的0.90~0.99倍), 模型性能基本保持稳定, 表明其对参数u的选择具有一定鲁棒性. 参数q用于调节异常判别的严格程度, 如图4(a)所示, 当q取值范围是$1E-5\sim 1E-2$时, 模型检测性能随着q的减小逐渐趋于稳定, 表明模型对q的设定具有良好的适应性.



(a) 危险系数q的影响 (b) 折扣率γ_0的影响

(c) 有效时延上界值d_{lim}的影响 (d) 检测间隔*Interval*的影响

Fig. 4 Impact of varying super-parameters on detection performance of CLV method

图4 不同超参数对CLV方法检测性能的影响

4 实验评估与结果分析

本节首先介绍实验环境以及采集的数据集，接着通过实验验证威胁模型中的观点以及 CLV 方法的可行性、有效性、可扩展性和鲁棒性.

4.1 实验环境与数据集描述

实验环境运行于 Ubuntu 20.04 系统的虚拟主机，物理主机的 CPU 为 Intel Core i7-11800H，核心数为 8，内存 32 GB. 实验使用 Mininet[34] 模拟智能体网络，使用交换机模拟 Agent worker，如图 5 所示. 在该网络中，正常链路的带宽设置为 100 Mbps，伪造链路的带宽设置为 50 Mbps，表明伪造链路和恶意 Agent worker 的引入会降低消息的传输和处理性能. 此外，分别基于 Scapy[35] 和 PyPcap[36] 实现不同性能的中继模块，基于 iPerf[37] 工具生成背景流量. 如表 2 所示，实验考虑了 6 种攻击场景，包括高或低的传输或转发时延、是否过载 Agent 协调器，以及是否存在背景流量.

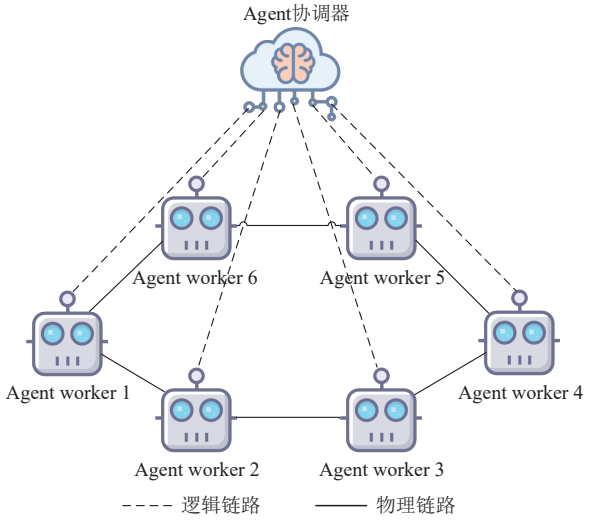


Fig. 5 Experimental topology

图 5 实验拓扑

Table 2 Parameter Settings in Varying Attack Scenarios

表 2 不同攻击场景下的参数配置

攻击场景	传输时延/ms	中继模块	转发时延	链路是否过载	背景流量	协调器过载攻击
Case 1	5	PyPcap	低	否	无	无
Case 2	0.5	Scapy	高	否	无	无
Case 3	0.5	PyPcap	低	是	35~45 Mbps	无
Case 4	0.5	PyPcap	低	否	无	无
Case 5	0.5	PyPcap	低	否	无	1 000 报文/s
Case 6	0.5	PyPcap	低	否	20~30 Mbps	1 000 报文/s

关于不同攻击场景的说明如下：Case 1 表示高传输时延，且无扰动 Agent 协调器和背景流量的情况. Case 2 表示高转发时延，且无扰动和背景流量的情况. Case 3 表示低传输和转发时延，无扰动但背景流量致使伪造链路过载的情况. Case 4 表示低传输和转发时延，且扰动和背景流量的情况. Case 5 表示低传输和转发时延，无背景流量但存在扰动的情况. Case 6 表示低传输和转发时延，且存在扰动和背景流量的情况. 特别地，Case 4、Case 5、Case 6 对应低时延网络环境下的攻击场景.

关于参数设置的说明如下：在 Case 1 中，传输时延设置为 5 ms，与文献 [14,16-17]，保持一致. 其他情况下传输时延设置均为 0.5 ms，以指示低时延的网络环境. 在 Case 2 中，相关方法[14,16,20] 基于 Scapy 实现中继模块，其相比 PyPcap 实现的中继模块具有更高的转发时延. 在 Case 3 中，背景流量设置为 35~45 Mbps，使得链路过载但不会导致中继模块崩溃. 同时，仅采集链路过载时的时延用于统计分析. 在 Case 5 中，设置攻击速率为每秒 1 000 报文，以扰动 Agent 协调器但避免其服务不可用. 在 Case 6 中，设置背景流量为 20~30 Mbps，以避免链路过载.

表 3 给出了实验数据集的描述. 其中，特征向量为 $(d_{cs}, d_{sc}, d_l, load)$，$d_{cs}$ 和 d_{sc} 分别表示 Agent 协调器到 Agent worker 以及 Agent worker 到 Agent 协调器之间的测量时延，d_l 和 $load$ 分别表示链路 l 的测量时延和负载. 相较于文献 [16]，此处增加了链路负载的特征，用以计算链路质量. 采样周期数为链路探测的次数，实际每隔 2 s 探测一次网络中的所有链路.

Table 3 Description of the Experimental Datasets

表 3 实验数据集描述

攻击场景	Agent 数量	伪造链路数量	采样周期数
Case 1~ 6	5	1	1 500
Case 4~ 6	5	1	1 500
Case 6	10, 20, ⋯, 100	1	6 000
Case 6	30	1, 3, 6, 9	6 000
样本特征	分类类别	类别比例	训练与测试比例
$(d_{cs}, d_{sc}, d_l, load)$	正常链路，伪造链路	3∶1	2∶1

4.2 威胁模型验证

该实验旨在验证威胁模型中提出的观点：1)在 Case 1~3 的攻击场景下，当链路传输时延较高，中继模块性能较低(亦即转发时延较高)或伪造链路过载时，伪造链路与正常链路之间的性能差异是显著的.

2)在 Case 4~6 的攻击场景下, 当攻击违背 Case 1~3 情况时, 伪造链路与正常链路之间的性能差异不再明显, 进而导致现有方法的检测性能不佳.

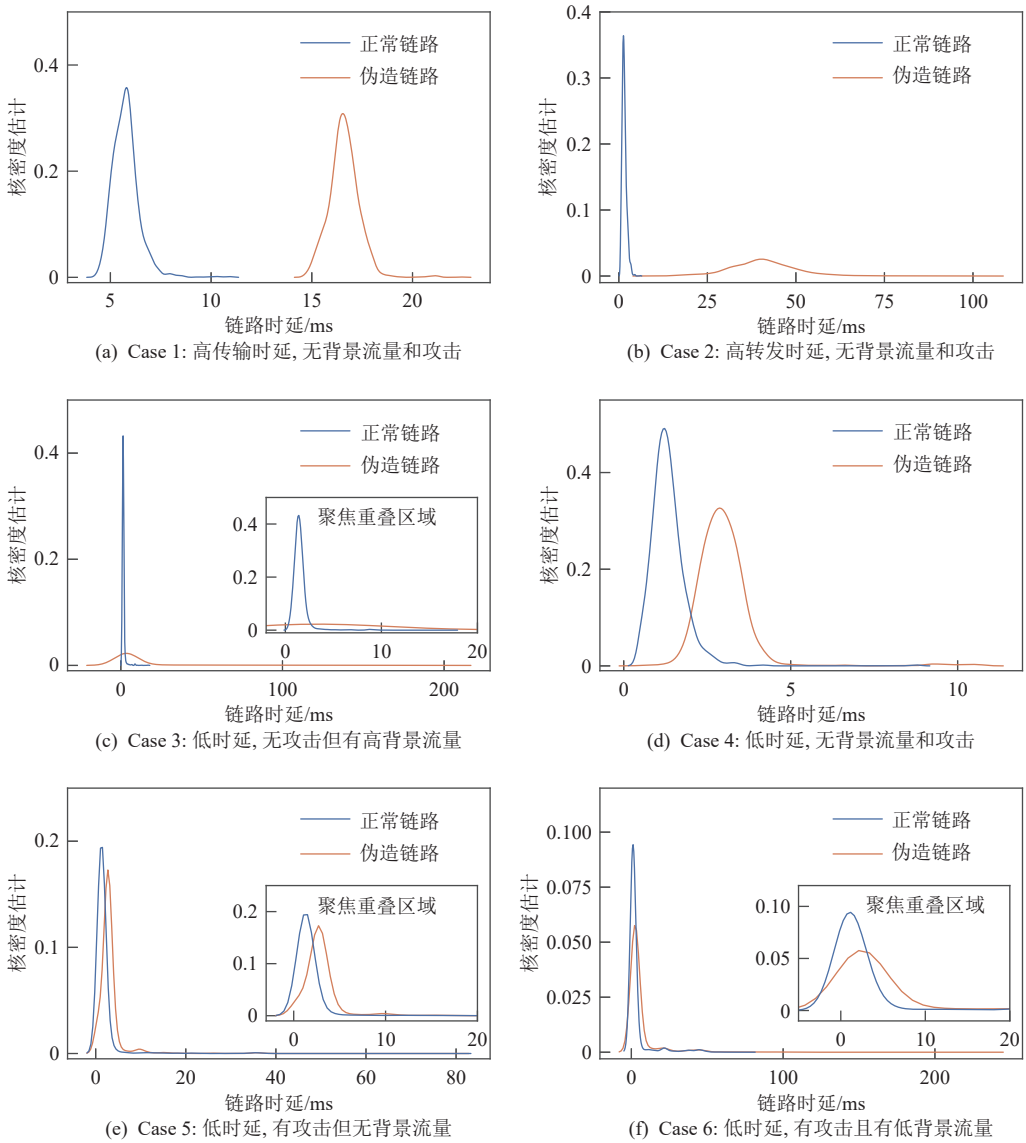
实验结果如图 6 所示. 在 Case 1~3 的攻击场景下, 正常链路与伪造链路上测量时延的分布差异是明显的, 相关工作[14,16,20]表明, 这些方法均能取得理想的检测效果. 然而, 在 Case 4~6 的攻击场景下, 正常链路与伪造链路上测量时延的分布差异不再明显. 根据 4.4 节中的实验结果可知, 此时相关方法均表现出较低的检测率和较高的误判率. 因此, 威胁模型中的观点得到了验证.

4.3 攻击检测方法的可行性验证

为了说明低时延网络中 CLV 方法的可行性, 需要验证 2 点: 1)对于关联链路而言, 伪造链路的存在是否会导致它与正常链路之间产生可观测的性能差异. 2)在低时延网络中, 本文使用的统计特征能否指示关联链路中的伪造链路.

首先, 对第 1 点进行验证. 为展示链路之间的性能差异, 采集多种攻击场景下 100 个采样周期的数据, 计算每个周期关联链路的链路质量, 如图 7 所示. 可见, 在所有场景下, 正常关联链路之间的性能差异非常细微, 而由正常链路和伪造链路构成的关联链路之间的性能差异相对明显且容易观测到. 因此, 第 1 点得到了验证.

接着, 对第 2 点进行验证. 考虑低时延的网络环境, 仅关注 Case 4~6 的攻击场景. 同时, 获取这些攻击场景下 100 个采样周期的数据, 利用多径传输模拟方法得到链路的轨迹序列, 并计算统计特征熵值和



(a) Case 1: 高传输时延, 无背景流量和攻击

(b) Case 2: 高转发时延, 无背景流量和攻击

(c) Case 3: 低时延, 无攻击但有高背景流量

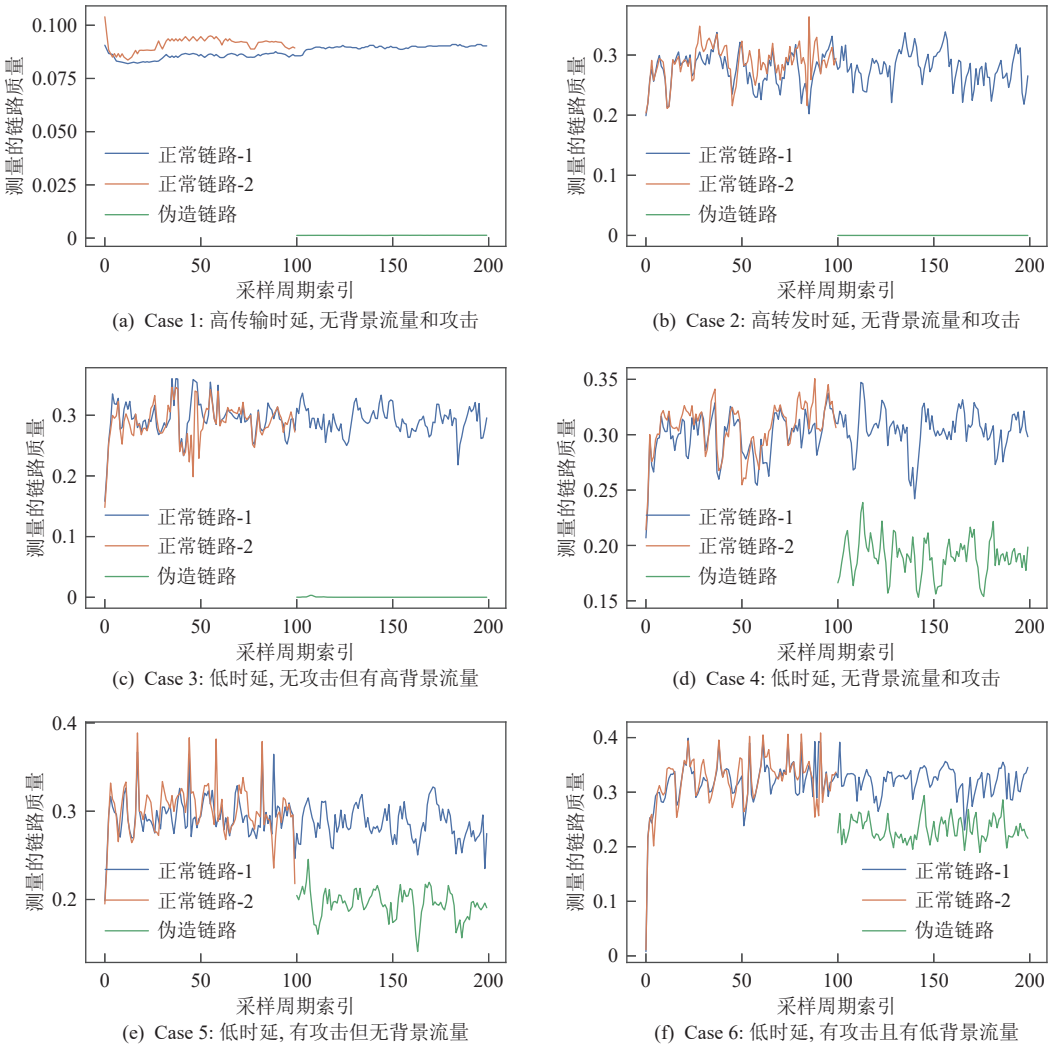
(d) Case 4: 低时延, 无背景流量和攻击

(e) Case 5: 低时延, 有攻击但无背景流量

(f) Case 6: 低时延, 有攻击且有低背景流量

Fig. 6 Distribution of measured delay before and after introducing fabricated link in varying attack scenarios

图 6 不同攻击场景下引入伪造链路前后测量时延的分布



(a) Case 1: 高传输时延, 无背景流量和攻击　(b) Case 2: 高转发时延, 无背景流量和攻击

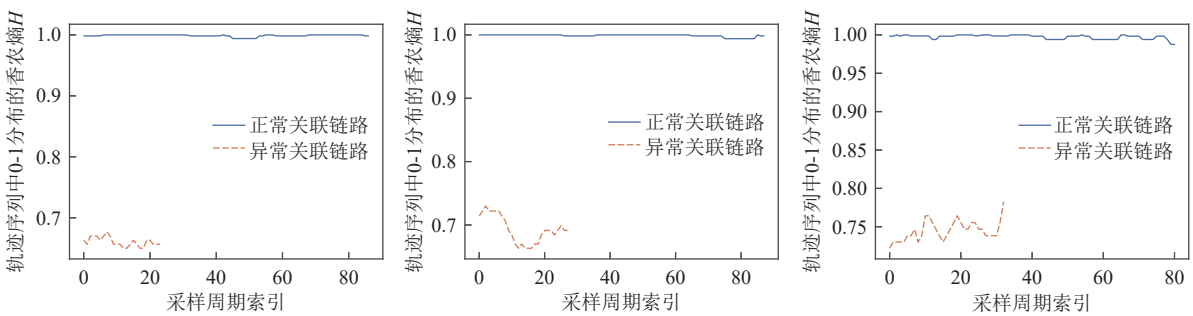
(c) Case 3: 低时延, 无攻击但有高背景流量　(d) Case 4: 低时延, 无背景流量和攻击

(e) Case 5: 低时延, 有攻击但无背景流量　(f) Case 6: 低时延, 有攻击且有低背景流量

Fig. 7　Time series of measured link quality under varying attack scenarios

图 7　不同攻击场景下测量的链路质量的时间序列

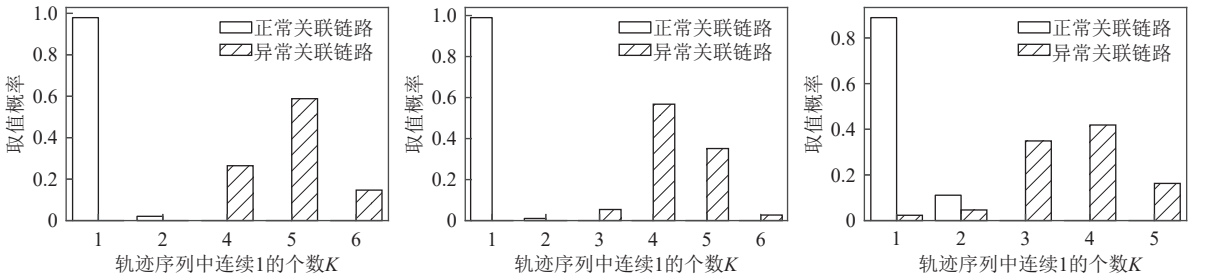
K值. 如图 8 所示, 对于正常关联链路, 轨迹序列中 0-1 分布的香农熵值在各个时刻均接近 1, 表明正常关联链路之间的性能差异非常细微. 而对于包含一条伪造链路的异常关联链路, 轨迹序列中 0-1 分布的香农熵值明显低于 1, 表明关联链路中正常链路与伪造链路之间存在可观测的性能差异. 如图 9 所示, 对于正常关联链路和异常关联链路, 其轨迹序列中连续 1 的数量分布不同, 且前者K值一般大于后者. 综上, 这 2 个统计特征能够用于指示伪造链路的存在, 故第 2 点得到了验证. 需要说明的是, 图 9 中二者的熵值序列长度不同, 这是因为它们的K值不同, 导致统计周期的大小不同, 如式(11)所示.



(a) Case 4: 低时延, 无背景流量和攻击　(b) Case 5: 低时延, 有攻击但无背景流量　(c) Case 6: 低时延, 有攻击且有低背景流量

Fig. 8　Time series of entropy values for trajectory sequences of correlated links under varying attack scenarios

图 8　不同攻击场景下关联链路轨迹序列熵值的时间序列



(a) Case 4: 低时延, 无背景流量和攻击 (b) Case 5: 低时延, 有攻击但无背景流量 (c) Case 6: 低时延, 有攻击且有低背景流量

Fig. 9 Distribution of K values for trajectory sequences of correlated links under varying attack scenarios

图 9 不同攻击场景下关联链路轨迹序列K值的分布

4.4 攻击检测方法的有效性验证

为了验证低时延网络中 CLV 方法的有效性, 获取 Case 4~6 攻击场景下采集的数据集, 对比 CLV 方法与相关工作中方法的检测效果, 评估指标包括检测率 *TPR*、误判率 *FPR*、检测准确率 *Pr* 和 *F*1 分数. 另外, 相关工作中检测方法的具体实现在威胁模型部分已经进行了介绍, 此处不予赘述. 需要说明的是, 该实验严格按照相关工作的描述进行参数设置和模型调优. 对于 RLV[16], 使用 sklearn 训练其使用的机器学习模型, 并采用网格搜索寻找最优的超参数. 对于 TopoGuard+[17], 无需设置超参数. 对于 TrustTopo[20], 设置超参数$c=0.5$使得模型的检测效果最佳.

表 4~6 给出了不同攻击场景下相关方法的检测结果, 从中可以发现, 相关方法无法兼顾检测率和误判率. RLV 相较于 TrustTopo 和 TopoGuard+具有更高的检测率和误判率. 其原因在于, 这些方法直接基于测量的时延训练检测模型, 难以得到能够准确识别伪造链路的阈值. 过高的阈值导致较低的检测率和误判率, 而过低的阈值导致较高的检测率和误判率. CLV-OPT 和 CLV-RAW 分别表示是否采用提出的数据处理方法. 经过转换的统计特征具有更清晰的界限, 并且阈值的确定过程具有良好的可解释性, 故针对该数据集, CLV 方法能够取得超过 99% 的检测率和约 0% 的误判率. 由于该数据集的规模较小, CLV-OPT 与 CLV-RAW 的性能差异并不明显, 在扩展性和鲁棒性验证的实验中, 二者的差异将会凸显出来.

Table 4 Comparison of Detection Performance for Relative Methods in Case 4 Scenario

表 4 Case 4 场景下相关方法的检测性能对比

检测指标	TopoGuard+[17]	TrustTopo[20]	RLV[16]				CLV	
			MLP	RF	KNN	SVM	RAW	OPT
TPR	4.00	56.9	62.72	58.72	60.72	58.18	99.30	99.00
FPR	0.30	12.2	17.30	13.98	16.93	12.76	0	0.40
Pr	93.02	82.34	78.54	81.34	78.83	82.56	100	100
*F*1	7.67	67.30	69.75	68.20	68.60	68.26	99.65	99.60

Table 5 Comparison of Detection Performance for Relative Methods in Case 5 Scenario

表 5 Case 5 场景下相关方法的检测性能对比

检测指标	TopoGuard+[17]	TrustTopo[20]	RLV[16]				CLV	
			MLP	RF	KNN	SVM	RAW	OPT
TPR	7.85	53.45	71.02	63.71	64.54	69.49	91.70	99.60
FPR	4.95	18.05	25.80	17.13	18.81	23.95	0	0
Pr	61.33	73.93	73.54	79.42	78.08	75.80	100	100
*F*1	13.92	62.04	72.26	70.71	70.67	72.18	95.65	99.80

Table 6 Comparison of Detection Performance for Relative Methods in Case 6 Scenario

表 6 Case 6 场景下相关方法的检测性能对比

检测指标	TopoGuard+[17]	TrustTopo[20]	RLV[16]				CLV	
			MLP	RF	KNN	SVM	RAW	OPT
TPR	8.95	54.00	64.84	64.74	62.96	68.30	98.70	99.10
FPR	5.75	20.15	25.33	25.60	25.40	28.73	0	0
Pr	60.88	72.83	71.96	72.07	71.67	70.81	100	100
*F*1	15.61	6 202	68.21	68.21	67.03	69.53	99.35	99.55

4.5 攻击检测方法的可扩展性验证

为了验证低时延网络中 CLV 方法的可扩展性, 仅考虑包含扰动和背景流量的 Case 6 场景. 以交换机模拟 Agent worker, 设置数量 10~100. 同时, 以 *AUC* 值为指标设置 CLV 方法中超参数的值. *AUC*是评估分类器性能的常用指标, 其值为 0~1, 且越接近 1 表明分类器的性能越好. 针对 CLV 方法, 超参数取值对检测性能的影响如图 4 所示.

图 4(a)显示了危险系数的影响, 它对应式(16)中的参数q_{h}和式(18)中的参数q_{k}. 尽管当危险系数低于1E−3时检测性能保持不变, 但为了取得尽可能低的误判率, 设置危险系数值为1E−5. 图 4(b)和图 4(c)分别显示了折扣率和有效时延上界的影响, 它们对应于式(10)中的参数γ_0和式(8)中的参数d_{lim}. 显然, 最佳的折扣率值为 0.9, 最佳的有效时延上界值为 5.

图 4(d) 表明了统计周期次数的影响. 根据 $T_{det}=Interval\times T_{sta}$ 知, 统计次数越少, 检测时延越小. 而统计次数超过 10 时检测性能保持不变, 故将其设置为 10.

不同网络规模下相关方法的检测性能如图 10 所示. 可以发现, 随着 Agent worker 数量的增加, 这些方法的检测性能总体呈现出不同程度的下降. 其中, TopoGuard+方法的检测性能变化较小, 始终保持在较低的状态, 这归因于其过高的检测阈值. 然而, 这也会导致过低的检测率. 因此, 它们难以适用于低时延网络. 相反, RLV 和 TrustTopo 方法具有较高的检测率和误判率, 这归因于其过低的检测阈值. 由于误判率过高, 如在大部分情况下超过 30%, 它们同样难以适用于低时延网络. 此外, CLV-RAW 相较于 CLV-OPT 性能下降得更快, 这表明数据处理方法能够有效缓解测量误差, 提高模型的可扩展性. 相较于这些方法, CLV 方法表现出更好的可扩展性和稳定性. 因此, CLV 方法的可扩展性得到了验证.

4.6 攻击检测方法的鲁棒性验证

在该文中, 鲁棒性特指检测方法对背景流量和扰动的容忍能力. 为了验证 CLV 方法的鲁棒性, 仅关注包含攻击和背景流量的 Case 6 场景, 并引入不同数量的伪造链路, 以及不同大小的链路传输时延. 需要说明的是, 为了确保一定强度的背景流量顺利生成, 限制 Agent worker 的数量为 32~48, 伪造链路的数量为 1~9. 当伪造链路的数量为 9 时, 受影响的 Agent worker 数量为 18, 这意味着网络中约一半的 Agent worker 会受到攻击的影响. 这种极端的场景有利于评估 CLV 方法的鲁棒性.

如图 11 和图 12 所示, 随着伪造链路数量的增加, CLV-RAW 和 CLV-OPT 方法的检测性能均会有所下降. 其原因在于, 随着伪造链路数量的增加, 网络波动更加剧烈, 导致采集到的无效数据增多, 进而影响模型的检测性能. 尽管如此, 当伪造链路的数量达到 9 时, CLV-RAW 仍能够取得约 95.61% 的检测率和约 3.7% 的误判率, CLV-OPT 方法可取得约 96.66% 的检测率和约 2.36% 的误判率. 实验结果表明, CLV 方法具有较强的抗扰动能力.

如图 13 和图 14 所示, 链路的传输时延会影响 CLV 方法的检测性能. 具体地, 链路传输时延越小, 伪造链路与正常链路的性能差异越小, CLV 方法的检测性能相对越差. 当链路传输时延为 0.1 ms 时, CLV-RAW 方法仅能取得 4.97% 的检测率, 而 CLV-OPT 依然能够取得约 95.23% 的检测率和约 0.28% 的误判率. 该实验结果表明, 所提出的数据处理方法能够有效缓解测量误差的影响, 进而增强模型的鲁棒性. 综上, CLV 方法的鲁棒性得到了验证.

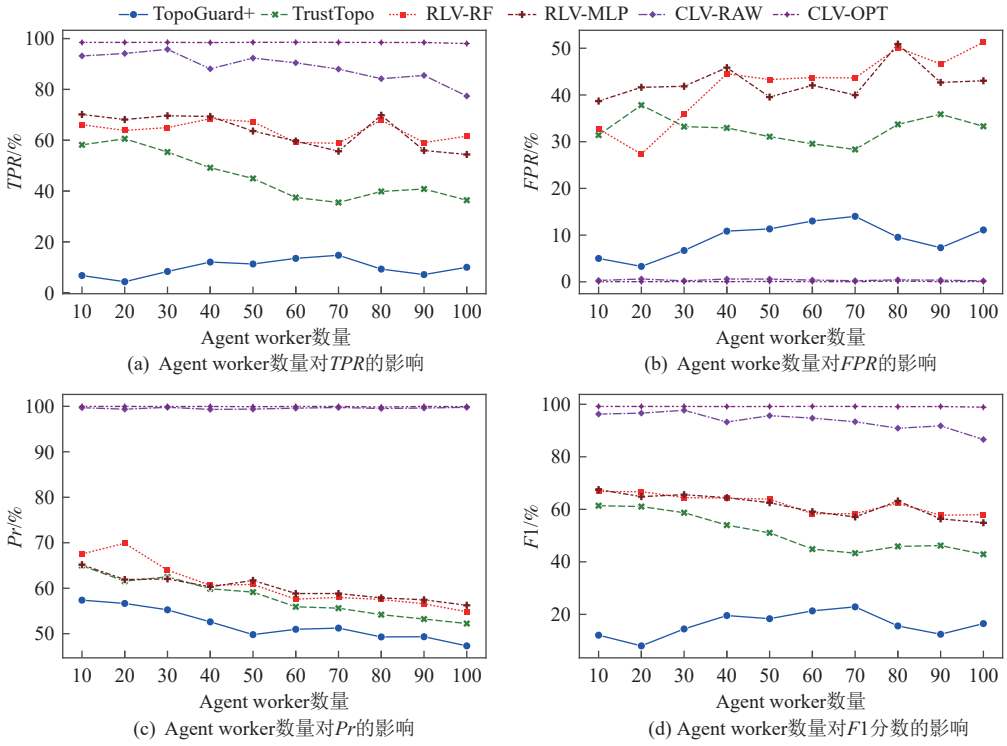


Fig. 10 Comparison of detection performance with related methods across varying network scales

图 10 不同网络规模下相关方法检测性能的对比

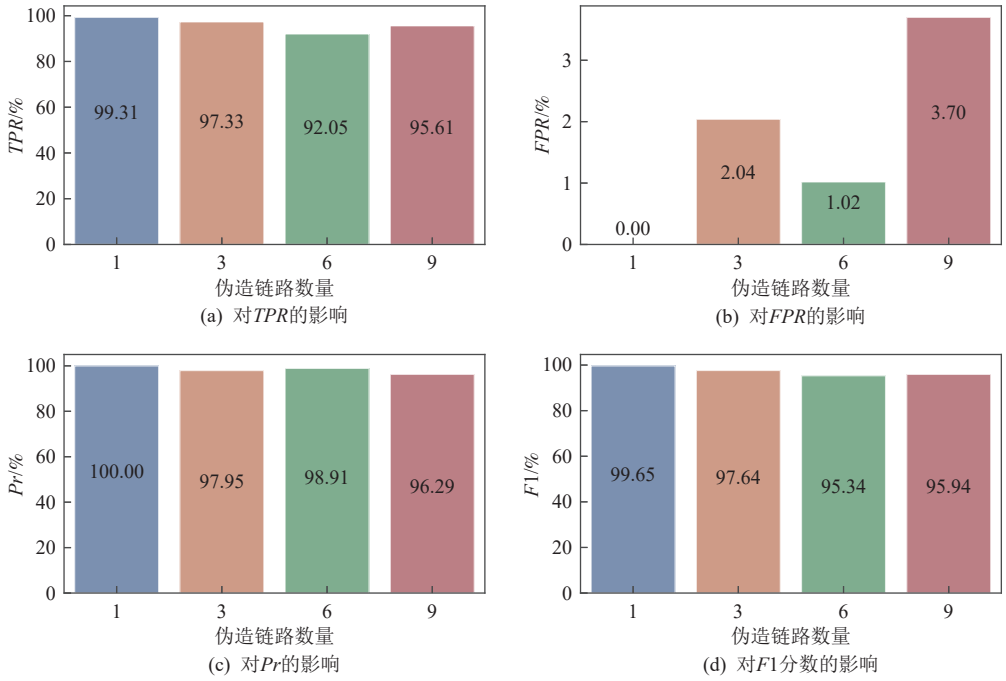


Fig. 11 Detection performance of CLV-RAW method under varying fabricated link counts

图 11 不同伪造链路数量下 CLV-RAW 方法的检测性能

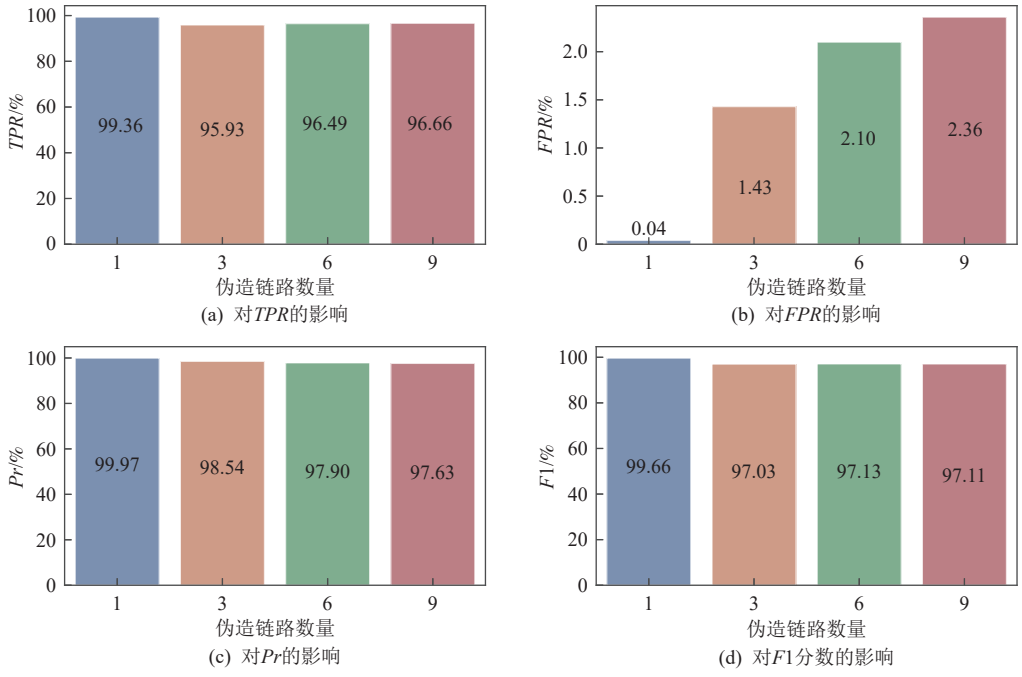


Fig. 12 Detection performance of CLV-OPT method under varying fabricated link counts

图 12 不同伪造链路数量下 CLV-OPT 方法的检测性能

5 结 论

本文针对低时延智能体网络中的伪造链路攻击检测问题，提出了一种基于关联链路验证的检测方法. 该方法通过引入数据预处理机制提升测量鲁棒性，利用多径传输模拟将关联链路间的性能差异转化为统计特征，并结合极值理论与概率分布拟合确定检测阈值，从而实现对伪造链路的有效识别. 为验证方法的有效性，本文从可行性、检测性能、可扩展性、鲁棒性等方面开展实验，结果表明 CLV 在不同网络规模和攻击环境下均具有良好的适应性和稳定性.

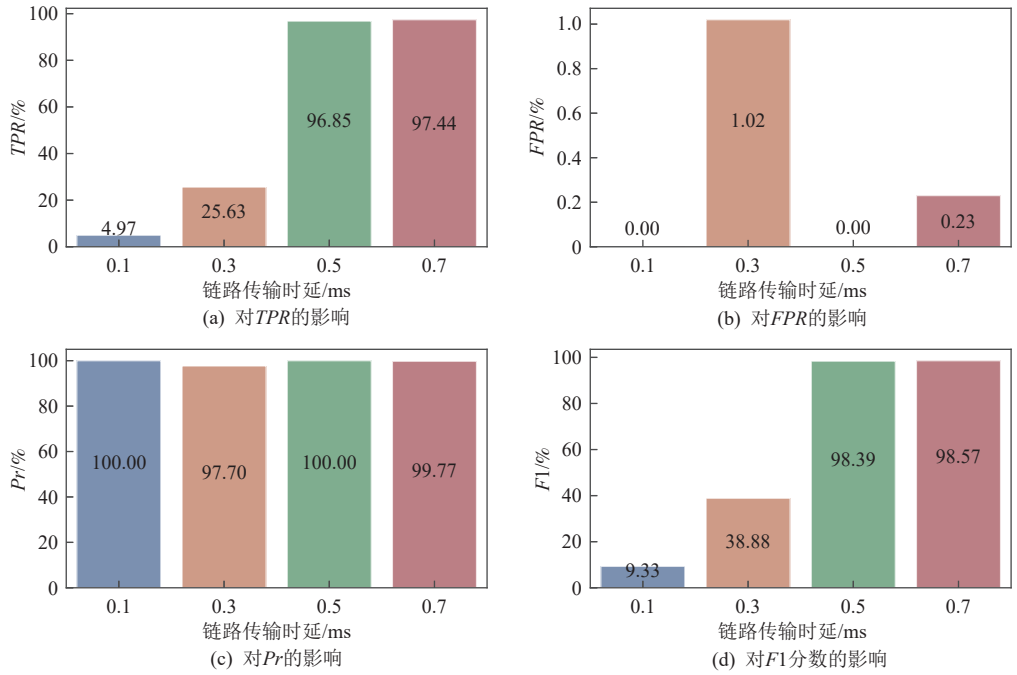


Fig. 13 Detection performance of CLV-RAW method under varying link delay

图 13 不同链路时延下 CLV-RAW 方法的检测性能

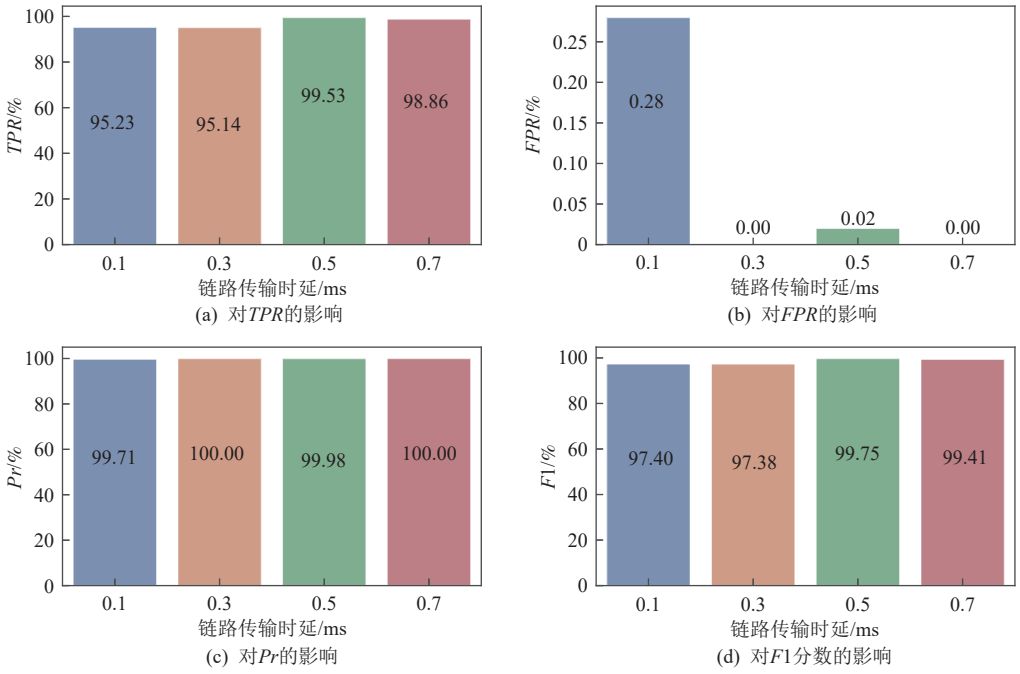


Fig. 14 Detection performance of CLV-OPT method under varying link delay

图 14 不同链路时延下 CLV-OPT 方法的检测性能

作者贡献声明：张峰、俸皓、刘玉明提出了智能体网络伪造链路攻击检测问题，设计了整体方案并撰写论文；周秀民、章茂淳、陈镇铭负责部分算法和实验.

参 考 文 献

[1] Liu Yan, Guo Bin, Li Nuo, et al. CrowdTransfer: Enabling crowd knowledge transfer in AIoT community[J]. IEEE Communications Surveys & Tutorials, 2024, 27(2): 1191−1237

[2] Liu Sicong, Guo Bin, Fang Cheng, et al. Enabling resource-efficient AIoT system with cross-level optimization: A survey[J]. IEEE Communications Surveys & Tutorials, 2023, 26(1): 389−427

[3] Chen Weize, You Ziming, Li Ran, et al. Internet of agents: Weaving a Web of heterogeneous agents for collaborative intelligence[C]//Proc of the Int Conf on Learning Representations (ICLR). San Diego, CA:

OpenReview. net, 2025: 1−32

[4] Agent Network Protocol Contributors. Agent Network Protocol (ANP). [EB/OL]. [2025-7-16]. https://github.com/Agent-network-protocol/AgentNetworkProtocol

[5] Fu Bo, Smith W, Rizzo D, et al. Robust task scheduling for heterogeneous robot teams under capability uncertainty[J]. IEEE Transactions on Robotics, 2022, 39(2): 1087−1105

[6] Soltani S, Amanlou A, Shojafar M, et al. Security of topology discovery service in SDN: Vulnerabilities and countermeasures[J]. IEEE Open Journal of the Communications Society, 2024, 5: 3410−3450

[7] Zheng Zheng, Xu Mingwei, Li Qi, et al. Defending against SDN network topology poisoning attacks[J]. Journal of Computer Research and Development, 2018, 55(1): 207−216 (in Chinese)
(郑正, 徐明伟, 李琦, 等. 网络拓扑污染攻击防御机制研究[J]. 计算机研究与发展, 2018, 55(1): 207−216)

[8] Taheri M, Khorasani K, Shames I, et al. Undetectable cyber attacks on communication links in multi-agent cyber-physical systems[C]//Proc of the 59th IEEE Conf on Decision and Control (CDC). Piscataway, NJ: IEEE, 2020: 3764−3771

[9] Ding Lifu, Yan Gangfeng. A survey of the security issues and defense mechanisms of multi-agent systems[J]. CAAI Transactions on Intelligent Systems, 2020, 15(3): 425−434

[10] Ding Lifu, Yan Gangfeng. A survey of security issues and defense mechanisms in multi-agent systems[J]. Information and Control, 2018, 47(6): 651−660 (in Chinses)
(丁礼富, 严刚峰. 多智能体系统安全性问题及防御机制综述[J]. 信息与控制, 2018, 47(6): 651−660)

[11] Cao Yinzhi, Yu A F, Aday A, et al. Efficient repair of polluted machine learning systems via causal unlearning[C]//Proc of the 2018 on Asia Conf on Computer and Communications Security (ASIACSS). New York: ACM, 2018: 735−747

[12] Zeng Zengri, Peng Wei, Zeng Detian. Improving the stability of intrusion detection with causal deep learning[J]. IEEE Transactions on Network and Service Management, 2022, 19(4): 4750−4763

[13] Smyth D, McSweeney S, O'Shea D, et al. Detecting link fabrication attacks in software-defined networks[C]//Proc of the 26th Int Conf on Computer Communication and Networks (ICCCN). Piscataway, NJ: IEEE, 2017: 1(1): 1−8

[14] Al Salti I, Zhang N. LINK-GUARD: An effective and scalable security framework for link discovery in SDN networks[J]. IEEE Access, 2022, 10(10): 130233−130252

[15] Soltani S, Shojafar M, Mostafaei H, et al. Link latency attack in software-defined networks[C]//Proc of the 17th Int Conf on Network and Service Management (CNSM). Piscataway, NJ: IEEE, 2021, 1(1): 187−193

[16] Soltani S, Shojafar M, Mostafaei H, et al. Real-time link verification in software-defined networks[J]. IEEE Transactions on Network and Service Management, 2023, 20(3): 3596−3611

[17] Skowyra R, Xu Lei, Gu Guofei, et al. Effective topology tampering attacks and defenses in software-defined networks[C]//Proc of the 48th Annual IEEE/IFIP Int Conf on Dependable Systems and Networks (DSN). Piscataway, NJ: IEEE, 2018: 374−385

[18] Marin E, Bucciol N, Conti M. An in-depth look into SDN topology discovery mechanisms: Novel attacks and practical counter-measures[C]//Proc of the 2019 ACM SIGSAC Conf on Computer and Communications Security, New York, ACM, 2019: 1101−1114

[19] Chou L D, Liu C C, Lai M S, et al. Behavior anomaly detection in SDN control plane: A case study of topology discovery attacks[J]. Wireless Communications and Mobile Computing, 2020, 2020(1): 8898949

[20] Huang Xinli, Shi Peng, Liu Yufei, et al. Towards trusted and efficient SDN topology discovery: A lightweight topology verification scheme[J]. Computer Networks, 2020, 170: 107119

[21] Shrivastava P, Agarwal A, Kataoka K. Detection of topology poisoning by silent relay attacker in SDN[C]//Proc of the 24th annual Int Conf on Mobile Computing and Networking. New York: ACM, 2018: 792−794

[22] Wang Jiadai, Liu Jiajia, Guo Hongzhi, et al. Deep reinforcement learning for securing software defined industrial networks with distributed control plane[J]. IEEE Transactions on Industrial Informatics, 2022, 18(6): 4275−4285

[23] Kong Dezhang, Shen Yi, Chen Xiang, et al. Combination attacks and defenses on SDN topology discovery[J]. IEEE/ACM Transactions on Networking, 2023, 31(2): 904−919

[24] Azzouni A, Boutaba R, Trang N T M, et al. sOFTDP: Secure and efficient OpenFlow topology discovery protocol[C]//Proc of 2018 IEEE/IFIP Network Operations and Management Symp. Piscataway, NJ: IEEE, 2018: 1−7

[25] Nehra A, Tripathi M, Gaur M S, et al. SLDP: A secure and lightweight link discovery protocol for software defined networking[J]. Computer Networks, 2019, 150: 102−116

[26] Jia Yongze, Xu Lei, Yang Yuwang, et al. Lightweight automatic discovery protocol for OpenFlow-based software defined network-ing[J]. IEEE Communications Letters, 2020, 24(2): 312−315

[27] Gu Yongpu, Li Dong, Yu Junqing. IM-OFDP: An improved openflow-based topology discovery protocol for software defined network[C]//Proc of 2020 IFIP Networking Conf (Networking). Piscataway, NJ: IEEE, 2020: 628−630

[28] Hauser F, Schmidt M, Haberle M, et al. P4-MACsec: Dynamic topology monitoring and data layer protection with MACsec in P4-based SDN[J]. IEEE Access, 2020, 8: 58845−58858

[29] Jero S, Koch W, Skowyra R, et al. Identifier binding attacks and defenses in software-defined networks[C]//Proce of the 26th USENIX Conf on Security Symp. Berkeley, CA: USENIX Association, 2017: 415−432

[30] Zhou Qizhao, Yu Junqing, Li Dong. A dynamic and lightweight framework to secure source addresses in the SDN-based networks[J]. Computer Networks, 2021, 193: 108075

[31] Meena R C, Bhatia S, Jhaveri R H, et al. HyPASS: Design of hybrid-SDN prevention of attacks of source spoofing with host discovery and address validation[J]. Physical Communication, 2022, 55: 101902

[32] Haan L, Ferreira A. Extreme Value Theory: An Introduction[M]. New York: Springer, 2006: 65−126

[33] Siffer A, Fouque P A, Termier A, et al. Anomaly detection in streams with extreme value theory[C]//Proc of the 23rd ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining (KDD). New York: ACM, 2017: 1067−1075

[34] Mininet Project Contributors. An instant virtual network on your laptop (or other PC)[EB/OL]. [2025-07-16]. https://mininet.org/

[35] Scapy community. Scapy: The Python-based interactive packet manipulation program & library[EB/OL]. [2025-7-16]. https://github.com/secdev/scapy

[36] Arturo Filastò. PyPcap - Python libpcap module. [EB/OL]. [2025-07-16]. https://github.com /pynetwork/pypcap

[37] ESnet/Lawrence Berkeley National Laboratory. iPerf-The ultimate speed test tool for TCP UDP and SCTP[EB/OL]. [2025-07-16]. https://iperf.fr



Zhang Feng, born in 1977. PhD. His main research interests include network security and data science.

张　峰, 1977 年生. 博士. 主要研究方向为网络安全、数据科学.



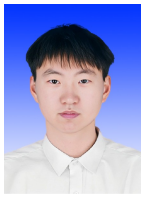
Feng Hao, born in 1978. PhD, associate professor. His main research interests include computing intelligence and distributed system.

俸　皓, 1978 年生. 博士, 副教授. 主要研究方向为计算智能、分布式系统.



Zhou Xiumin, born in 2000. Master candidate. His main research interests include agentic network and AIoT.

周秀民, 2000 年生. 硕士研究生. 主要研究方向为智能体网络、人工智能物联网.



Zhang Maochun, born in 2001. Master candidate. His main research interests include artificial intelligence and agentic network.

章茂淳, 2001 年生. 硕士研究生. 主要研究方向为人工智能、智能体网络.



Chen Zhenming, born in 2000. Master candidate. His main research interests include artificial intelligence and agentic network.

陈镇铭, 2000 年生. 硕士研究生. 主要研究方向为人工智能、智能体网络.



Liu Yuming, born in 1991. PhD. His main research interests include software-defined networking, programmable networks, and network security.

刘玉明, 1991 年生. 博士. 主要研究方向为软件定义网络、可编程网络、网络安全.