

基于动态浏览器指纹的链接检测技术研究

高凡, 王健, 刘吉强

(北京交通大学智能交通数据安全与隐私保护技术北京市重点实验室, 北京 100044)

摘要: 随着用户隐私泄露和网络欺诈问题的出现, 传统检测机制已经不能阻挡愈演愈烈的攻击行为, 需要新的技术手段辅助进行 Web 安全检测。作为沟通用户与网络数据的桥梁, 浏览器广泛应用于用户与各种应用程序之间的交互, 其中浏览器指纹包含用户浏览器及设备等多种特征信息, 其独特性可以极大地提高用户识别的准确率。但是浏览器指纹随时间不断发生变动, 基于静态指纹的研究方案难以满足动态指纹检测的要求。针对这一问题, 为了提高动态浏览器指纹的识别和链接检测能力, 从多维角度获取参数并进行筛选, 得到细粒度和高区分度的特征进行指纹检测, 防止过多不必要参数带来的运算代价, 并将双向门控循环单元 (BiGRU, bidirectional gating recurrent unit) 引入指纹检测工作中, 提出了基于堆叠 BiGRU 的动态浏览器指纹链接检测模型, 解决了指纹检测过程中忽略前后向信息交互导致的检测精度不高的问题, 在准确率和链接时长等指标上有了提升。进而针对不同特征参数的重要性不同, 赋予相同权重会造成检测误差的问题, 结合注意力机制聚焦稳定性更强的指纹特征, 提出了基于 Att-BiGRU 的动态浏览器指纹链接检测模型, 较堆叠 BiGRU 有了检测能力上的优化, 进一步提高了浏览器指纹的检测准确率, 并在与基准模型的综合指标对比结果中表现出了良好的链接能力。

关键词: 浏览器指纹; 动态链接; 攻击检测; 注意力机制; Web 安全

中图分类号: TP393

文献标识码: A

DOI: 10.11959/j.issn.2096-109x.2022049

Research on link detection technology based on dynamic browser fingerprint

GAO Fan, WANG Jian, LIU Jiqiang

Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing 100044, China

Abstract: With the emergence of user privacy leakage and online fraud, traditional detection mechanisms can no longer stop the intensified attacks and new methods are needed to assist in web security detection. As a bridge between users and network data, browsers are widely used in the interaction between users and various applications. The browser fingerprint contains various characteristic information of the user's browser and device, and its uniqueness can greatly improve the accuracy of user identification. However, browser fingerprints may change over

收稿日期: 2021-11-25; 修回日期: 2022-03-04

通信作者: 王健, wangjian@bjtu.edu.cn

基金项目: 国家重点研发计划 (2020YFB2103800)

Foundation Item: The National Key R&D Program of China (2020YFB2103800)

引用格式: 高凡, 王健, 刘吉强. 基于动态浏览器指纹的链接检测技术研究[J]. 网络与信息安全学报, 2022, 8(4): 144-156.

Citation Format: GAO F, WANG J, LIU J Q. Research on link detection technology based on dynamic browser fingerprint[J]. Chinese Journal of Network and Information Security, 2022, 8(4): 144-156.

time, and solutions based on static fingerprints cannot meet the requirements of dynamic fingerprints detection. Motivated by this challenge, it's necessary to improve the recognition and link detection capabilities of dynamic browser fingerprints. Besides, parameters from multi-dimensional perspective need to be obtained and screened to get fine-grained and high-discrimination features for model detecting, preventing the calculation cost caused by too many unnecessary parameters. Bidirectional Gating Recurrent Unit (BiGRU) was introduced into the fingerprint detection, and a stacked BiGRU detection model based on dynamic browser fingerprints was proposed. It solved the problem of low detection accuracy caused by ignoring the interaction of forward and backward information in the process of fingerprints detection, and it also had a certain improvement in indicators such as accuracy and linking time. Furthermore, in view of the different importance of different feature parameters, the same weight given to parameters will result in detection error. Combined with the attention mechanism to focus more stable fingerprint features, an Att-BiGRU fingerprint detection model based on dynamic browser fingerprints was proposed. It optimized the detection ability compared to BiGRU, further improved the detection accuracy of browser fingerprinting, and showed good linking ability on the comprehensive index under the benchmark model.

Keywords: browser fingerprint, dynamic linking, attack detection, attention mechanism, Web security

0 引言

科学技术的发展促进着互联网的不断革新,网络为人类社会提供的便捷服务深刻改变了人们信息交互的方式。浏览器作为 Web 应用的入口,是安全防护的薄弱环节,尤其作为向用户提供服务的工具,攻击者通过浏览器采用恶意手段窃取用户隐私。例如, Cookie 曾被用来解决用户频繁登录同一网站时加载速度过慢的问题,弥补了无状态 HTTP (HyperText transfer protocol) 的局限性,但也常被广告商用来获取用户的偏好信息^[1-2]。Web 开发者为了抵御各类安全威胁,通常根据用户 IP (Internet protocol) 地址拦截特定访问,但 IP 易被修改和伪造。因此,需要一个有效的安全检测方法,在提高互联网安全性的同时将认证过程对用户的影响降到最低,防止恶意用户访问。

浏览器指纹主要取决于自身的软硬件参数进行识别,是每个用户独一无二的数据库。将浏览器指纹作为识别手段可以极大地提高用户的识别率^[3],增强在线账户的安全性,如可以应用到银行等网站中,通过在登录时验证用户的浏览器指纹,系统可以轻松阻止来自未知设备的未授权访问。浏览器指纹也可以应用于反欺诈等风控场景,由于收集到的属性之间存在众多依赖关系,因此可用来检查指纹是否已被篡改,或者是否与指纹所属的设备相匹配,由此区分普通用户与欺诈用户。

但是浏览器的不断更新、用户自行更改设备

属性选项或者使用了一系列反跟踪软件,均会导致浏览器指纹不断发生变化,针对静态指纹信息的解决方案可能在一段时间后就丧失了研究价值,因此需要更有效的方法进行变动指纹的识别。

综合上述现状及问题,本文的主要工作内容如下。

1) 从多维角度获取浏览器指纹特征信息,减少特征单一造成的检测误差,并在分析后选取其中区分度高、细粒度和稳定性强的属性,防止过多不必要参数带来的运算代价。

2) 将双向门控循环单元 (BiGRU, bidirectional gating recurrent unit) 引入指纹检测工作中,提出基于动态浏览器指纹的堆叠双向门控循环单元检测模型,通过前后序相互依赖提高检测的准确率,进而提升链接动态浏览器指纹的能力。

3) 考虑特征权重问题,使用注意力机制调整模型关注点,聚焦阈值更高的特征属性,提出 Att-BiGRU 模型适用于浏览器指纹的链接工作,减少噪声污染,在综合表现上进一步提高浏览器指纹的检测能力。

1 研究现状

1.1 浏览器指纹

浏览器指纹识别首先由电子前沿基金会 (EFF, Electronic Frontier Foundation) 提出, ECKERSLEY

等^[4]在大量浏览器样本中收集了 470 161 个指纹,发现浏览器指纹可以更精确地显示设备的多样性。在浏览器指纹的概念出现之后,特征属性由原先的 Cookie 扩充到了 User-Agent、Canvas 等参数,浏览器指纹的部分特征示例如表 1 所示。

浏览器指纹常被用于多种目的,Laperdrix 等^[5]将浏览器指纹的使用分为两大类。

1) 负面应用:未知的第三方通过伪造浏览器指纹假冒用户获取隐私信息,或者通过识别已知漏洞攻击其设备。

2) 正面应用:通过实时检测和更新进行用户身份识别,防止恶意用户欺诈,还可以通过验证设备为系统增强在线服务的安全性。

现有的研究工作主要聚焦于更改浏览器属性特征以达到欺瞒跟踪者的目的,进而防止攻击^[6]。对于浏览器指纹带来的恶意跟踪问题,Torres 等^[7]使用 FP-Block 的解决方案分离 Web 身份,浏览器连接不同站点时会呈现不同的指纹,从而防止跟踪;Fiore 等^[8]根据以往的数据创建相似指纹,呈现伪造的一致性指纹对抗恶意检测,但这两类研究的覆盖范围均不完整。Laperdrix 等^[9]将噪声引入 Canvas、AudioContext API 等特征中,并随机化 JavaScript 对象的枚举顺序以增加浏览器的不确定性;Wang 等^[10]将拉普拉斯噪声注入低秩矩阵进行指纹的隐私保护,减少指纹跟踪的影响,但这类方式的缺点是其他特征仍旧可以发挥作用,甚至特殊化参数在某种程度上使指纹更加突出。

通过修改指纹特征、引入噪声或者使用 Tor

浏览器^[11]的做法在某种程度上可以免受恶意跟踪的困扰,但只能提供有限的保护,难度较大且会增加用户操作的复杂度。因此,需要进行浏览器指纹的安全检测研究工作,并对浏览器指纹的识别验证现状做进一步介绍。

1.2 指纹识别验证

浏览器指纹是增强认证的一个有效手段,如 Bursztein 等^[12]设计的基于画布指纹识别的名为 Picasso 的解决方案,不仅可以过滤非正常流量,还可以检测出设备的浏览器和操作系统系列,查看指纹与实际设备之间是否存在不匹配的现象。Faiz-rhademi 等^[13]提出的 FPGuard 方法,能够在运行时检测和预防浏览器指纹,通过收集并分析与指纹活动有关的细粒度数据,进而识别 Web 服务及在线行为。

2019 年,有两项相似的研究通过画布指纹识别增强身份验证。Rochet 等^[14]为每个设备创建个性化模型,并使用深度学习算法为用户建立二进制分类模型,在得分高于设置的特定阈值后才能验证。而 Laperdrix 等^[15]在研究中要求当前浏览器必须能够渲染出与先前完全相同的 Canvas,否则就会阻止当前设备匹配,用户只能使用其他方式进行识别。但这两项研究对特征的精度匹配要求很高,同时特征训练和提取不全面,识别不了变动指纹。

如今一些从事在线安全检测的企业已经使用浏览器指纹来增强身份验证,如 SecurAuth^[16]作为自适应访问控制解决方案的提供商,成功地构建了通过设备指纹识别的基于启发式的身份验证

表 1 指纹特征部分示例
Table 1 The examples of fingerprint features

特征	值
User-agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.190 Safari/537.36
Platform	Win32
Timezone	-480
Vendor	Google Inc.
List of fonts (JS)	Arial, Arial Black, Arial Narrow, Bahnschrift, Calibri and 69 others
Do Not Track	NC
Canvas	7d3176108470111c1c95d5536fe69384b57df861
WebGL Renderer	ANGLE (AMD Radeon(TM) R5 Graphics Direct3D11 vs_5_0 ps_5_0)
List of plugins	Plugin 0: Chrome PDF Plugin; Portable Document Format; internal-pdf-viewer. Plugin 1: Chrome PDF Viewer; mhjfbmdgcfjbbpaojofohoeofgiehjai. Plugin 2: Native Client; internal-nacl-plugin

系统, 该系统也使用了浏览器指纹作为多因素身份验证过程的一部分。Lovation 公司拥有名为 ClearKey^[17]的解决方案, 该解决方案将设备信息的收集作为其多因素身份验证框架的一部分进行集成, 提供了自己的方法来处理带有模糊逻辑算法的指纹检测。

浏览器指纹技术能够辨别用户的访问设备, 在不增加用户操作且不降低用户体验的情况下增强认证和访问的安全性。但需要注意的是, 浏览器指纹的特征数据时常会发生变化, 在一定的变化范围之内, 为了能够判别出新的指纹与已知库中的指纹是否来源于同一浏览器, 需要进一步研究动态浏览器指纹的链接工作。

1.3 指纹动态链接

指纹是用户设备和环境的直接反映, 随着系统更新或配置更改, 浏览器指纹会经常发生改变, 因此需要具备了解这些变化并链接动态指纹的能力。

指纹变化的原因主要分为以下几点。

- 1) 自动变化: 如浏览器或内置插件自动升级更新。
- 2) 环境改变: 如进入不同的时区。
- 3) 用户自定义设置: 用户更改某些选项, 如清除 Cookie、更换字体或禁用 DNT 等。
- 4) 使用反跟踪操作: 如使用某些扩展程序混淆指纹, 或者给指纹的某些特征增加噪声 (如 FPRandom^[9]中为 Canvas 和 AudioContext 引入了用户无法察觉的噪声), 使指纹表现出了随机性。

Eckersley^[4]首次提出了指纹变动问题, 使用算法将浏览器指纹特征进行直接比较, 估算待测指纹和原先指纹的演变关系, 但该研究方法人为干预因素过多, 不稳定且准确率过低。2016 年, Alaca 等^[19]使用设备指纹进行身份验证时, 考虑稳定性研究了指纹的差异接受度, 由登录系统决定是否接受版本更新或者字体被修改的设备, 但只是简单地做了介绍, 并没有进行深入探讨和研究。

2018 年, Vastel 等^[19]描述了浏览器指纹变化的频繁程度, 引出静态规则与机器学习中的随机森林相结合的算法 FPStalker, 追踪浏览器指纹的演变过程。在此基础上, Li 等^[20]提出通过

将新指纹链接到上一个指纹来提高浏览器指纹的稳定性, 使用长短期记忆网络模型来学习浏览器指纹不断变动的潜在关系。Li 等^[21]在 Gomez-Boix 工作的基础上, 首次对数百万个指纹数据进行了大规模的研究, 分析了浏览器指纹的隐私和安全问题, 并强调研究动态浏览器指纹的重要性。

结合以上研究工作, 考虑到指纹特征之间的依赖关系和动态指纹的链接问题, 本文对动态浏览器指纹进行检测技术研究。

2 基于动态指纹的链接检测

2.1 特征筛选

浏览器指纹属性繁多, 在检测用户并链接浏览器实例的过程中, 过多的特征参数会引入噪声, 导致实验的鲁棒性偏低, 同时会增加不必要的运算代价。因此, 从设备、操作系统、浏览器和用户配置等多维角度获取浏览器指纹特征属性并分析, 提取出符合以下要素的特征。

(1) 可区分度高

尽量选择区分度比较高的特征, 至少能从数千台设备中标识某一设备, 即使指纹被仿冒也能够从中检测出来, 增强安全性。因此引入信息熵^[22]的概念来衡量指纹特征的区分度, 信息熵可以作为参数信息的量化指标, 特征参数越复杂, 不同浏览器指纹之间出现相同情况的概率越小, 信息熵越大。浏览器指纹特征熵值示例如表 2 所示。

表 2 浏览器指纹特征熵值示例
Table 2 The examples of feature entropy of browser fingerprint

特征	熵值	特征	熵值
User-Agent	6.83	Accept	9.23
Plugins	3.13	Timezone	5.21
Fonts	3.59	Color Depth	3.88
CPU	0.14	Audio	2.58
Canvas	12.75	WebGL	17.07
Language	7.69	Vendor & Resolution	15.07

(2) 细粒度

浏览器特征中包含一定的冗余信息, 因此特征提取时需要对冗余信息进行过滤, 对粗粒度信

息进行分割,提取细粒度的数据作为特征参数。

例如, GPU 和 navigator.vendor 中均包含供应商等相关信息,需要在比较后筛除重复数据; User-Agent 包含的信息量较大,需要使用 User-Agent 解析库对其进行分解,得到浏览器、操作系统和是否为 PC 端等信息; Canvas、WebGL 等字符串表示复杂,需要使用哈希杂凑进行处理; Use of DNT 等特征参数简单,可以直接数值化操作等。

(3) 稳定性

在考虑浏览器指纹时常变化的情况下,需要保留一部分信息量低但稳定性较强的特征属性。针对浏览器指纹发生变化后产生的一系列问题,对指纹属性值中的变化进行分析,选取合理的特征。例如, Use of LocalStorage、Use of SessionStorage 等参数,用户基本不会对其改动,这些特征作为保留数据可以在检测初期帮助筛除不匹配的指纹数据。

在考虑上述 3 种要素的基础上,最终提取出 User-Agent、Accept、Cookies、Language、Canvas 等 25 种特征进行后续实验。

2.2 基于堆叠 BiGRU 的动态指纹链接检测

考虑到浏览器指纹的动态链接中有时间序列关系,经过分析后选用双向循环神经网络进行浏览器指纹检测:通过前序影响后序判断特征的相似度从而将待测指纹链接到现有指纹链中,通过后序影响前序判断链接的合理性。考虑到过拟合及运算代价问题,使用双向门控循环单元进行算法的改进,提出基于堆叠 BiGRU 的动态浏览器指纹检测模型,提高模型的鲁棒性,减少特征顺序对模型的影响,引入 Dropout 算法提高模型的泛化能力。基于堆叠 BiGRU 的浏览器指纹链接流程如图 1 所示。

2.2.1 指纹筛选

一个用户的全部浏览器指纹应构成一条指纹链,但浏览器指纹会随配置更改不断发生变动,为了更好地链接浏览器指纹的动态变化过程,同一指纹链需存在多条指纹数据进行训练。因此,筛除指纹数据库中链长度小于 6 的浏览器指纹,并对指纹链接进行初步的规则限制:同一指纹链中的所有指纹数据需要保持操作系统和浏览器类

型的稳定;对 Use of LocalStorage、CookiesEnable、Use of DNT 等特征的设置需相同;Canvas 值保持一致;Fonts 参数信息有交集;User-Agent、Vendor、Plugins、Language、Accept 等特征变化在一定相似性范围内等。

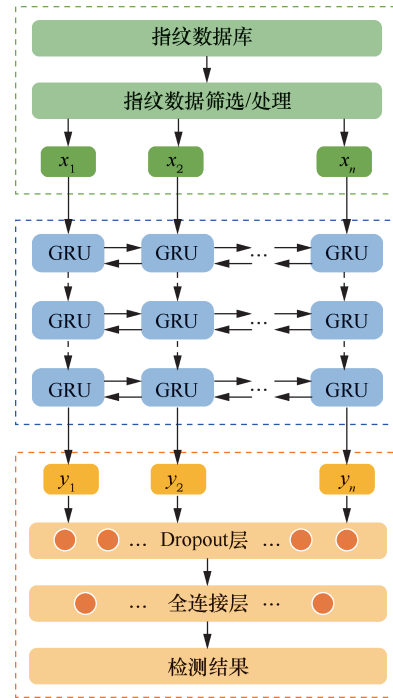


图 1 基于堆叠 BiGRU 的浏览器指纹链接流程

Figure 1 The flow of browser fingerprint link based on stacked BiGRU

在以上规则的基础上,将待测指纹与指纹数据库中的已知指纹进行对比,如算法 1 所示。

算法 1 指纹匹配规则算法

输入 原始指纹数据集 F

输出 匹配的指纹 id 或者筛选后的指纹数据

集 F_{sub}

开始

Function:

Rules={rule1,rule2,...}

$F_{\text{sub}} \leftarrow \phi$

$F_{\text{pre}} \leftarrow \phi$

for $f_k \in F$ do

if verifyRules(f_k, f_u , Rules) then

if NumberofDiff = 0 then

$F_{\text{pre}} \leftarrow F_{\text{pre}} \cup \langle f_k \rangle$

else

$F_{\text{sub}} \leftarrow F_{\text{sub}} \cup \langle f_k \rangle$

```

    end if
  end if
end for
if  $|F_{\text{pre}}| > 0$  and sameIds( $F_{\text{pre}}$ ) then
  return  $F_{\text{pre}}[0].\text{id}$ 
else if  $|F_{\text{sub}}| > 0$  then
  return Fuction_BiGRU
else
  return generateNewId()
end if
end function

```

其中, F 为原始指纹数据集, f_u 为待测的未知指纹, f_k 为指纹库中某个待比较的浏览器指纹, F_{sub} 为粗略筛选后的数据集, F_{pre} 为精确匹配的数据集合。

在检验待测指纹 f_u 是否属于已知指纹库中的某条指纹链时, 先在已有规则条件下遍历指纹库中的每条指纹数据。如果符合规则, 则判断未知指纹 f_u 和已知指纹 f_k 的相似度, 如果二者之间完全匹配, 则将该指纹置于精确匹配的 F_{pre} 集合; 否则将其置于候选指纹集合 F_{sub} 。倘若精确匹配的 F_{pre} 集合中仅存在 1 条指纹数据, 则直接返回该指纹数据的 id; 如果 F_{pre} 集合中指纹数量大于 1, 则比较集合中所有指纹的 id 是否一致, 相同时直接返回历史 id, 表明待测指纹 f_u 属于该浏览器实例, 可以成功链接到指纹链上。但如果是候选集合 F_{sub} 中存在指纹数, 则进入 BiGRU 模型中继续判断。如果待测指纹完全不符合现有的筛选规则, 则说明指纹数据库中不存在待测指纹的历史数据, 需要生成一条新的指纹链, 并赋予其新的指纹 id。

2.2.2 双向门控循环单元

浏览器指纹数据特征繁多, 且随时间和配置更改等因素变动较大, 考虑到时间序列及单向网络会受到指纹数据顺序影响, 采用可以减少过拟合风险的双向门控循环单元进行指纹检测。

双向门控循环单元相较于单向门控循环单元增加了从后向前传输数据的隐藏层, 每一次输入都包含正向时间步和逆向时间步的 GRU, 输出由两个方向的 GRU 同时决定。

BiGRU 输出层的计算需要获取当前时间步 t 的输入 X_t , $t-1$ 时刻前向的隐藏状态 $\vec{H}_{t-1}$ 和反向隐藏状态 $\vec{H}_{t-1}$, 并通过对两个方向的隐藏状态加权求和得到在 t 时刻的隐藏状态 H_t , 并最终获取输出 Y_t , 具体计算式如式(1)~式(4)所示。

$$\vec{H}_t = \sigma(X_t W_{xh} + \vec{H}_{t-1} W_{hh} + b_h) \quad (1)$$

$$\vec{H}_t = \sigma(X_t W_{xh} + \vec{H}_{t-1} W_{hh} + b_h) \quad (2)$$

$$H_t = \vec{H}_t W_{xh} + \vec{H}_t W_{hh} + b_h \quad (3)$$

$$Y_t = H_t W_{hq} + b_q \quad (4)$$

其中, W_{xh} 、 W_{hh} 、 W_{hq} 为权重系数, b_h 、 b_q 为某时刻隐藏状态对应的偏差。

将 BiGRU 应用到动态浏览器指纹检测中, 首先, 进行规则筛选后的指纹链均含有至少 6 条浏览器指纹数据, 将当前时间步 t 时刻的指纹链以从后向前 (即从新到旧) 的顺序进行两两比较, 设 X_t 为 $\{x_1, x_2, \dots, x_n\}$, 指的是 t 时刻的待测指纹 f_u 与已知指纹 f_{t-1} 之间各个特征属性值的相似性结果, 设 X_{t-1} 指的是指纹 f_{t-1} 与指纹 f_{t-2} 之间属性值的相似性结果, 设 X_{t-2} 指的是指纹 f_{t-2} 与指纹 f_{t-3} 之间属性值的相似性结果, 将 $\{X_{t-2}, X_{t-1}, X_t\}$ 作为特征输入向量。其次, 引入堆叠 BiGRU 模型, 在本文实验中 3 层效果比单层和双层 BiGRU 明显, 超过 3 层难以训练。堆叠 BiGRU 的每一隐藏层包含不同的神经元, 并对每一层细胞的记忆状态进行重置。将第一层时间步的隐藏输出作为第二层时间步的隐藏输入, 以此类推, 将最后一个隐藏层的输出作为输出向量。将经过指纹筛选之后得到的候选集合 F_{sub} 处理后放入堆叠 BiGRU 模型中进行训练, 进而判断待测指纹 f_u 与 F_{sub} 集合中数据的相似度。

2.2.3 Dropout 算法

为了进一步减少模型过拟合的发生, 达到正则化的效果, 本文使用 Dropout 算法, 减轻神经元之间的共适应性。

Dropout 是通过随机忽略的方式来减少隐藏节点之间的相互作用, 在前向传播的过程中使具有一定概率 p 的神经元停止工作, 从而在训练过程不会太依赖局部特征, 使复杂神经网络在训练非大容量数据集时也能很好防止过拟合并提高模

型的效率, Dropout 算法示意如图 2 所示。

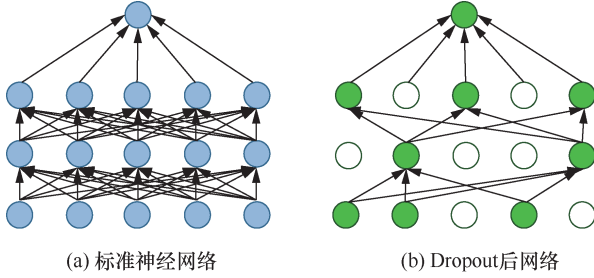


图 2 Dropout 算法示意

Figure 2 Schematic diagram of Dropout algorithm

在模型的训练阶段, 标准神经网络的计算公式如下:

$$y = f(w_i x + b_i) \quad (5)$$

设置概率 p , 则引入 Dropout 算法的神经网络计算式:

$$r_i = \text{Bernoulli}(p) \quad (6)$$

$$y = f(w_i(r_j \cdot x) + b_i) \quad (7)$$

其中, Bernoulli 为伯努利随机分布函数, 通过概率 p 随机生成一个 0-1 概率向量 w_i , r_i 表示网络的权重参数, b_i 为偏差。在模型测试阶段, 每个神经元的权重参数均需乘以 p 。

Dropout 算法提升了模型的精度, 使神经网络稀疏化, 减轻了部分特征的强依赖关系, 学习更加鲁棒的特征。本文研究在进行指纹检测过程时, 将 Dropout 的概率值设置为 0.3, 更好地提高模型的鲁棒性, 有效防止过拟合。

2.2.4 结果输出

在全连接层使用 Sigmoid 激活函数激活, 根据输出的概率与阈值比较, 判断该待测指纹 f_u 是否属于已知的指纹链。高于阈值则为同一条指纹链, 否则不属于已知的指纹链, 需要创建新的浏览器实例。

在算法 2 指纹匹配规则的基础上, 后续伪代码描述如算法 2 所示。

算法 2 基于堆叠 BiGRU 的指纹链接算法

输入 筛选后的指纹数据集 F_{sub}

输出 匹配的指纹 id

开始

Function:

candidates $\leftarrow \phi$

```

for  $f_k \in F_{\text{sub}}$  do
     $\langle X_{t-2}, X_{t-1}, X_t \rangle = \text{featureVector}(f_u, f_k)$ 
     $p \leftarrow P_{\text{BiGRU}}(f_u.\text{id} = f_k.\text{id} | \langle X_{t-2}, X_{t-1}, X_t \rangle)$ 
    if  $p \geq \lambda$  then
        candidates = candidates  $\cup \langle f_k, p \rangle$ 
    else
        return generateNewId()
    end if
end for

if |candidates| > 0 and sameIds(candidates)
then
    return candidates[0].id
else
    return generateNewId()
end if
end function

```

在得到规则筛选后的初步指纹数据集 F_{sub} 后, 遍历该数据集中的全部指纹数据, 并提取之前时刻的已知指纹与待测指纹构建特征向量, 经过堆叠 BiGRU 模型检测后得出概率与 λ 比较, λ 值设置为 0.994。如果 $p \geq \lambda$, 则加入 candidates 候选集中, 否则生成一条新的指纹链并赋予新的指纹 id。如果 candidates 中存在指纹数据并且数据来源相同, 则返回历史指纹 id, 将待测指纹链到现有指纹链上; 如果 candidates 集合中没有数据或数据来自不同指纹链, 则重新构建浏览器实例并赋予其新的指纹 id。

2.3 实验分析

2.3.1 实验环境

本实验在配置 Tensorflow 环境的 Pycharm professional 中进行, 实现配置如表 3 所示。

2.3.2 数据集

本文实验数据集起初通过开源的 Fingerprintjs2 自行获取, 但由于浏览器指纹的特征提取会侵犯用户的隐私, 因此数据获取只能在周边小范围内进行。同时时间特征持续检测的周期要求比较长, 持续时间不足会导致实验结果缺乏可行性, 考虑到最终收集到的指纹数据量很小(仅有 84 个用户参与采集过程)、持续时间比较短的现实因素, 结合开源数据集进行后续实验。

表3 实验配置
Table 3 The experimental configuration

实验环境	配置说明
操作系统	Win X64 系统
内存	16 GB
硬盘	447 GB
CPU	4 核
处理器	AMD A8-7100
Python 版本	3.6.12

由于上述提到的隐私问题,关于浏览器指纹的开源数据集很少,最终本文选用 Github 上的开源数据,并对其进行相应处理。该数据集的收集时间长达两年,共有近 2 000 个志愿者参与,公开数据集共 15 000 条。同时,由于实验需要对浏览器实例进行检测,因此对数据集中的每条数据标注 id 序列,相同的 id 表明来源于同一指纹链,经过筛选不符合规则及指纹长度小于 6 的数据后,最终用于实验的数据集共 4 803 条浏览器指纹。

2.3.3 实验结果分析

本文的基准模型为首次提出动态指纹概念的 Eckersley 算法,结合随机森林的 FPStalker 算法以及改进的 LSTM 算法,由于需要检测动态浏览器指纹的链接情况,本实验首先从拆分链数量、链接时长和所有权比率指标进行评估。

(1) 拆分链数量

期望将同一用户同一浏览器的所有指纹数据链接到同一链上,但现实会由于更改配置等,指纹差距过大被拆分为多个链,这种情况下,拆分链越少意味着链接检测效果越好,因此将拆分链数量作为评价指标之一。拆分链数量对比如图 3 所示。从图 3 可以看出, Eckersley 算法将指纹链分裂成多条,甚至在初始能够达到 14 条左右,稳定性比较差; LSTM 比 Eckersley 在初始值有所降低,但伴随时间的增长会反超 Eckersley 成为拆分数量最多的模型; FPStalker 在对比实验中表现最好,数目基本稳定在两条;而本文提出的堆叠 BiGRU 模型产生的拆分链数量比 Eckersley 和 LSTM 少,多于 FPStalker,为 4 条左右。

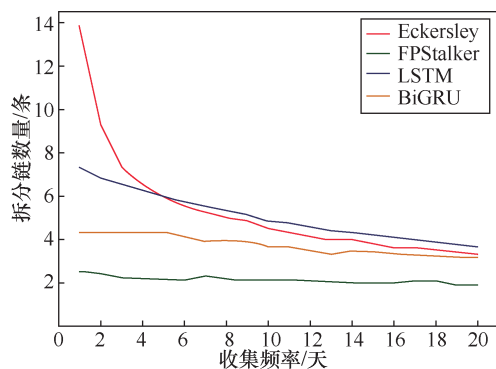


图3 拆分链数量对比

Figure 3 The comparison of the number of split chains

(2) 链接时长

链接时长表明能够追踪到的指纹时间。链接时间越长表明指纹动态变化产生的副作用越小,能够持续追踪到指纹的变化过程,将变化后的浏览器指纹链接到原有指纹链上。

在拆分链的基础上,同一条浏览器指纹链可能被拆分为多条子链,本文将最长拆分链的链接时间作为衡量数据进行对比。例如,用户 A 的第一条链 A1-A2-A3 为两个时间间隔(即 2T),第二条链 A4-A5-A6-A7 为 3 个时间间隔(即 3T),则最长链接时间为 3T,以此类推。如图 4 所示, BiGRU 模型的最长链接时间优于 LSTM 和 Eckersley,但与 FPStalker 相比有所差距。

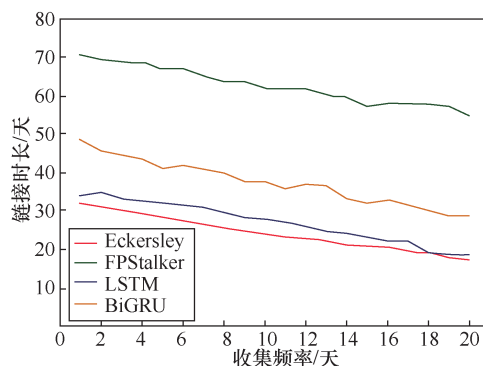


图4 最长链接时长对比

Figure 4 The comparison of maximum linking time

(3) 所有权比率

所有权意味着一条指纹链上是否只含有同一用户的指纹数据,如果掺杂其他用户的指纹,说明指纹链被污染,所有权比率将会下降,因此将其作为评价指标之一。即指纹链 A 中只有用户 A 的指纹数据,则 A 的所有权比率为 1,但如果错

误地将用户 B 的指纹数据链接到 A 指纹链上, 则 A 的所有权比率会下降。例如, A 指纹链中有 100 条指纹数据, 但是其中只有 90 条是 A 自身的正确数据, 其余 10 条来源于 B 或 C 等其他指纹链, 那么 A 的所有权比率为 $90/100=0.90$ 。

所有权比率对比如图 5 所示, 可以看出 LSTM 在所有权指标中表现最好, 链接出错的概率最低; 堆叠 BiGRU 的效果次之, 但随时间的增加趋近于 LSTM 的概率走向; FPStalker 对比 LSTM 和 BiGRU 的所有权比率有所下降, 表明在检测过程中会将部分指纹数据错误匹配; Eckersley 由于是最早提出应用于动态指纹检测的算法模型, 结构比较简单, 在链接过程中错误率最高。

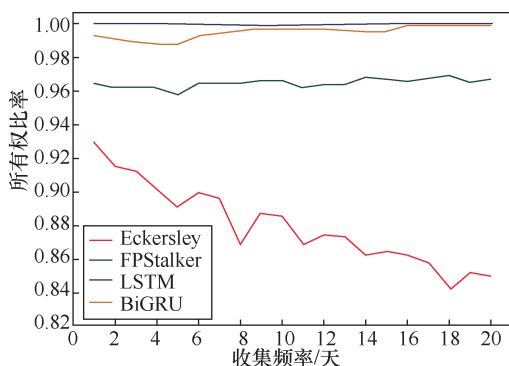


图 5 所有权比率对比

Figure 5 The comparison of ownership rate

3 动态指纹链接检测算法优化

3.1 基于 Att-BiGRU 的动态指纹链接算法

浏览器特征种类繁多, 但不同特征参数对指纹检测工作的贡献程度不同, 如 Canvas 及 Accept 等, 由于这些特征参数的信息熵较高, 不同指纹之间特征的相似度低, 因此可以赋予更高的权重。而 Use of LocalStorage 及 DNT 等特征可以在检测初期进行筛选, 但不能很好地进一步区分指纹信息, 因此赋予较小的权重。本文进一步提出基于注意力机制的 BiGRU 模型适用于浏览器指纹的链接工作, 聚焦区分度和阈值高的特征参数, 赋予特征不同的概率偏重, 提高模型检测的精确度。

Att-BiGRU 模型主要分为特征数据输入层、隐含层以及输出层。而隐含层包含 BiGRU 层、注意力机制(Attention)层及 Dense 层, 基于 Att-BiGRU 模型的浏览器指纹链接流程如图 6 所示。

首先, 在模型的输入层对浏览器指纹数据进行筛选, 指纹筛选流程如上文所示, 将待测指纹 f_u 与指纹库中的数据 f_k 进行匹配, 得到筛选后的指纹数据集 F_{sub} , 并将指纹数据处理为特征向量形式。其次, BiGRU 由前向 GRU 和反向 GRU 构成, 将上一时间步隐藏层状态 h_{t-1} 传输到下一隐藏层状态, 将时间序列关联起来, 并对输入特征向量进行更深层次的训练和提取。

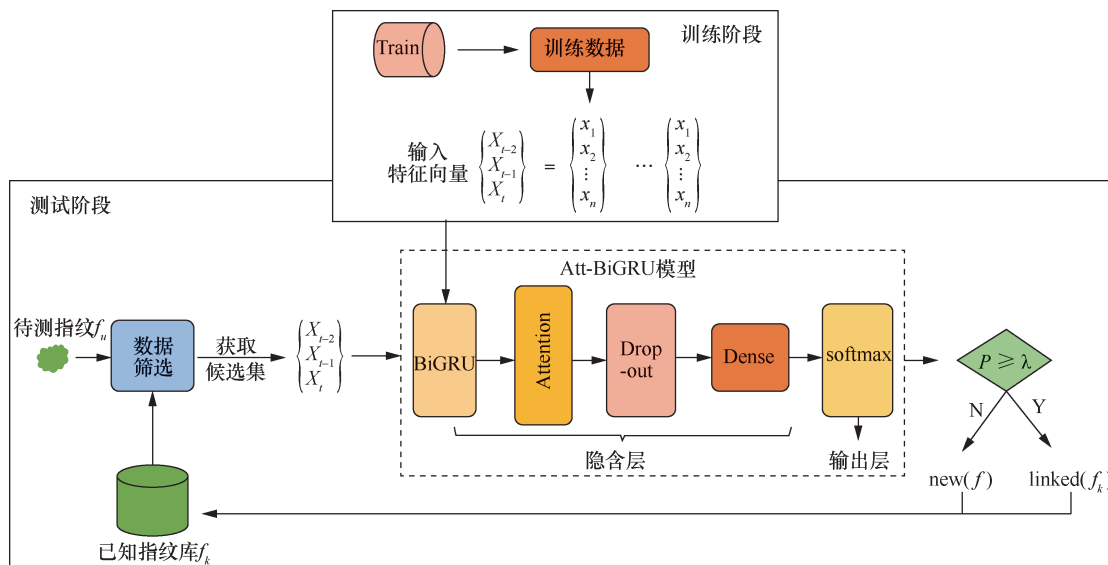


图 6 基于 Att-BiGRU 模型的浏览器指纹链接流程

Figure 6 The flow chart of browser fingerprint link based on Att-BiGRU model

使用注意力机制专注于区分度更高和唯一性更强的指纹特征数据,对 BiGRU 产生的隐藏层输出赋予对应的概率权重进行后续的应用。调整模型的关注点,并对其进行加权平均,即得到每个时间步隐藏状态的隐含表示 u_t ,再将其与初始化注意力矩阵 u_w 相乘,通过 softmax 函数获取到归一化权重 a_t ,最后将隐藏状态值 h_t 与获取到的权重 a_t 求和得到注意力机制表示 y ,计算公式如下。

$$u_t = \tanh(w_w h_t + b_w) \quad (8)$$

$$a_t = \text{softmax}(u_t^T u_w) \quad (9)$$

$$y = \sum_{i=1}^T a_i h_i \quad (10)$$

其中, w_w 为权重系数, b_w 为误差偏置系数, $\tanh$ 和 softmax 是采用的激活函数。

Dense 层将上层输出的指纹信息经过非线性变化提取特征之间的关联,并通过全连接层映射到输出空间。在 Attention 层和 Dense 层之间添加 Dropout 算法随机选取隐藏层神经元节点,进一步防止过拟合,增强指纹检测模型的鲁棒性,同时减少计算量。在全连接层使用 adam 优化器,利用 softmax 函数进行计算得到最终值,计算公式如下。

$$y' = M \cdot y \quad (11)$$

$$p = \text{softmax}(y') = \frac{\exp(y')}{\sum_{i=1}^T \exp(y')} \quad (12)$$

其中, M 为模型输出层的权重矩阵。将通过 softmax 得到的概率分布与判断待测指纹是否属于已知指纹链的二分类阈值 λ 进行比较, p 大于等于阈值则返回历史指纹 id, 否则生成新的浏览器指纹链并赋予新的指纹 id。

3.2 实验分析

本文实验对浏览器指纹在准确率和损失变化、拆分链数量、链接时长和所有权比率指标上进行模型对比。

(1) 准确率和损失变化

Att-BiGRU 指纹检测模型的训练集和测试集在准确率结果中随着训练次数的增加呈现出明显的上升趋势。如图 7 所示,该模型在前 7 轮次时的训练和测试效果不如堆叠 BiGRU 模型,但是随

着轮次增加,测试集效果有了明显提高。Att-BiGRU 模型训练集的准确率在该轮次内最高为 0.983 1,测试集的准确率最高为 0.996 0,相较 BiGRU 模型有所提升,训练效果良好。

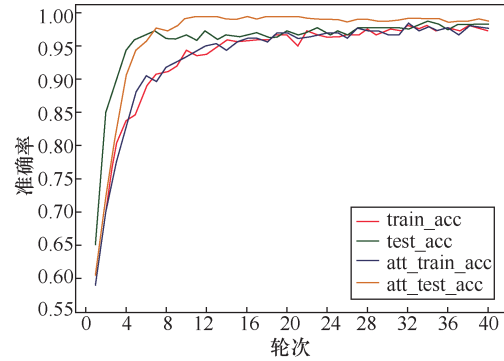


图 7 Att-BiGRU 模型准确率变化情况
Figure 7 The change of the accuracy of Att_BiGRU model

同时,在损失变化表现中呈现明显的下降趋势,如图 8 所示。其中 Att-BiGRU 模型训练集的损失变化最低为 0.057 3,测试集的损失变化最低为 0.026 0。

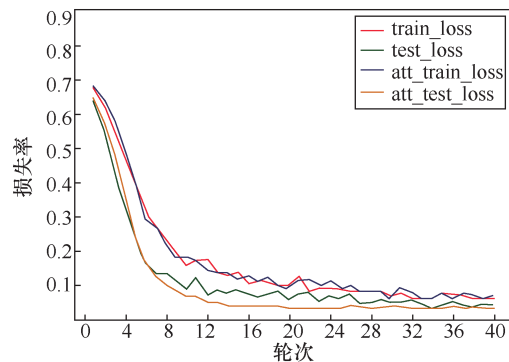


图 8 Att-BiGRU 模型损失变化情况
Figure 8 The loss change of Att-BiGRU model

(2) 拆分链数量

在拆分链数量的比较中,图 9 显示引进注意力机制的 Att-BiGRU 模型比堆叠 BiGRU 模型效果更好,考虑到注意力机制能够关注更重要的特征参数,减少了噪声污染,能够将指纹链拆分为更少的数量,处于 3~4 条的状态区间。

(3) 链接时长

最长链接时间即浏览器指纹拆分链中最长子链的持续时长。如图 10 所示,Att-BiGRU 模型的最长链接时间优于堆叠 BiGRU 和 LSTM,仅次于 FPStalker。

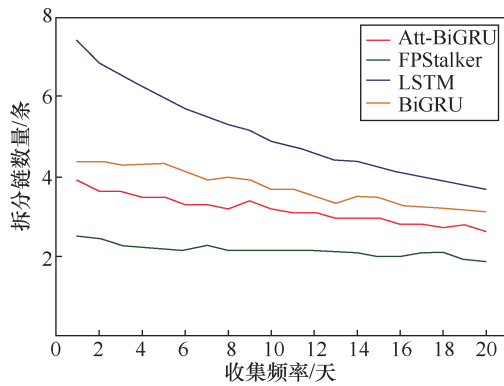


图 9 拆分链数量对比

Figure 9 The comparison of the number of split chains

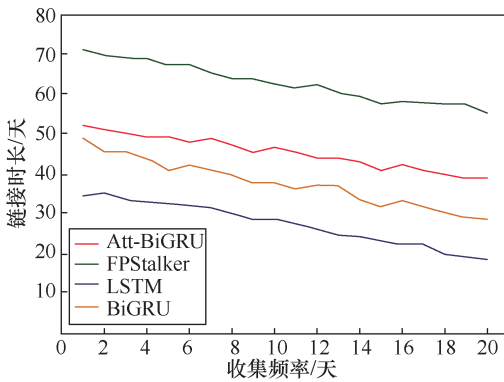


图 10 最长链接时间对比

Figure 10 The comparison of maximum linking time

(4) 所有权比率

所有权比率代表指纹链的纯净程度，用来评估是否仅包含相同用户的指纹数据。如图 11 所示，Att-BiGRU 模型的所有权比率相比 BiGRU 有一定提升，说明指纹链被污染的概率降低，识别变动指纹时出现混杂数据的情况有所改善，并随着时间的增长逐渐接近 LSTM 的比率。

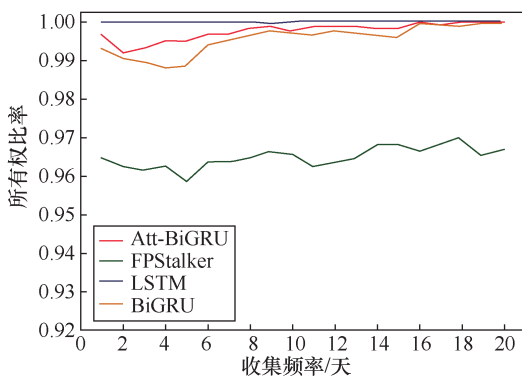


图 11 所有权对比

Figure 11 The comparison of ownership

(5) 运行时间

在进行模型运算时，指纹链长度和模型本身的复杂程度等导致运行时间不断攀升，如图 12 所示。其中，FPStalker 由于划分大量决策树导致运行时间最久，堆叠 BiGRU 检测模型由于深度网络计算，运行时间多于 LSTM；由于复杂模型的应用，Att-BiGRU 模型的运行时间比 BiGRU 稍久，但总体仍优于 FPStalker。

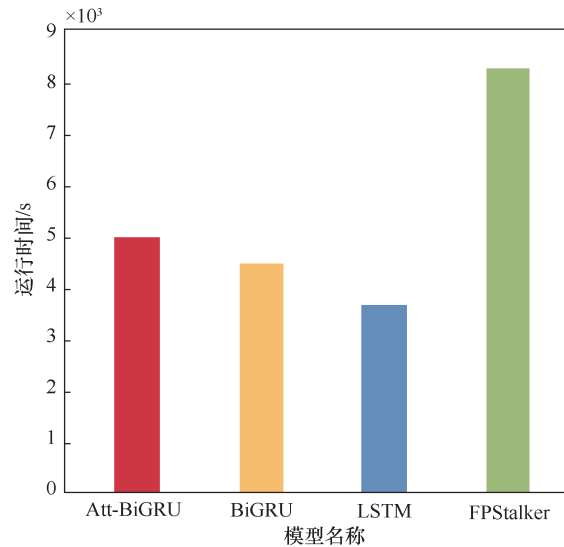


图 12 运行时间对比

Figure 12 The comparison of running time

3.3 整体分析

在引入注意力机制的 Att-BiGRU 模型中，准确率和损失变化较之前模型有所改善，同时能够将指纹链拆分为更少的数量，提高了追踪动态浏览器指纹的链接时间。虽然在拆分链数量和链接时长指标上没有 FPStalker 算法表现突出，但考虑到 FPStalker 将部分指纹数据链接到了不匹配的指纹链中，准确性相对较低。本文提出的堆叠 BiGRU 和 Att-BiGRU 模型在所有权指标中表现比较稳定，尽管稍低于 LSTM，但 LSTM 在链接时长和拆分链数量的评估指标中表现不佳。综合衡量，本文提出的基于动态浏览器指纹的检测模型在各个指标中的整体表现最为优越，Att-BiGRU 模型较 BiGRU 模型有一定提升，在模型训练和指纹检测实验中准确率更高。虽然本文实验尽所能地注意到运算代价的问题，但是为了提高链接时长及准确率，引入的 BiGRU 模型及注

意力机制等算法会提升运算代价, 因此运行时间有待改进。

4 结束语

本文针对浏览器指纹在更改配置或受到外界干扰后会发生变动的问题, 提出了基于堆叠 BiGRU 的动态浏览器指纹链接检测模型, 通过前后序相互依赖识别不断变动的指纹数据, 提升链接动态指纹的能力。并进一步使用注意力机制对指纹特征赋予不同权重, 关注区分度和信息熵更高的参数, 提出 Att-BiGRU 指纹链接检测模型。实验结果表明, Att-BiGRU 模型较堆叠 BiGRU 模型更关注关键信息, 减少了噪声污染, 在综合表现上进一步提升检测能力。

本文工作还有很多可以探讨和精进的空间。首先, 本文实验过程中涉及的数据只包含 PC 端信息, 后续可以引入移动端数据进行扩展研究。其次, 本文工作并没有从真正意义上实现跨浏览器指纹识别, 需要更多硬件设备上的辅助信息进行更深入的指纹检测。最后, 在指纹检测过程中, 可以考虑将指纹数据转化为图像进行链接。研究过程中做了一部分将 csv (comma-separated values) 中的指纹数据转化为图像的工作, 但由于预处理过程中丢失了很多参数信息, 不能很好地衡量链接能力, 需要探讨其他方案。

参考文献:

- [1] PAPADOPOULOS P, KOURTELLIS N, MARKATOS E P. Cookie Synchronization: everything you always wanted to know but were afraid to ask[C]//World Wide Web Conference. 2019: 1432-1442.
- [2] XUEHUI H, NISHANTH S. Characterising third PARTY cookie usage in the EU after GDPR[C]//ACM Conference on Web Science. 2019: 137-141.
- [3] ANDRIAMILANTO N, ALLARD T, GUELVOUIT G L. "Guess who?" large-scale data-centric study of the adequacy of browser fingerprints for web authentication[J]. Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2021, 7: 161-172.
- [4] ECKERSLEY P. How unique is your web browser[C]//Proceedings of the 2010 Privacy Enhancing Technologies Symposium. 2010: 1-18.
- [5] LAPERDRIX P, NATALIJA B, BENOIT B, et al. Browser fingerprinting: a survey[J]. ACM Trans, 2020, 14(2): 1-33.
- [6] ANDRIAMILANTO N, ALLARD T, GUELVOUIT G L. FPSelect: low-cost browser fingerprints for mitigating dictionary attacks against web authentication mechanisms[C]//Annual Computer Security Applications Conference (ACSAC '20). 2020: 627-642.
- [7] TORRES C, HUGO J, SJOUKE M. FP-Block: usable Web privacy by controlling browser fingerprinting[C]//Proceedings of the 20th European Symposium on Research in Computer Security, 2015: 3-19.
- [8] FIORE U, ANIELLO C, ALFREDO D, et al. Countering browser fingerprinting techniques: constructing a fake profile with Google Chrome[C]//Proceedings of the 17th International Conference on Network-Based Information Systems. 2014: 355-360.
- [9] LAPERDRIX P, BENOIT B, VIKAS M. FPRandom: randomizing core browser objects to break advanced device fingerprinting techniques[C]//Proceedings of the 9th International Symposium on Engineering Secure Software and Systems. 2017: 97-114.
- [10] WANG C, DANI J, LI X, et al. Adaptive fingerprinting: website fingerprinting over few encrypted traffic[C]//Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy (CODASPY '21). 2021: 149-160.
- [11] Saito. Tor fingerprinting: Tor browser can mitigate browser fingerprinting[C]//International Conference on Network-Based Information Systems, 2017: 504-517.
- [12] BURSZTETN E, ARTEM M, TADEK P, et al. Picasso: lightweight device class fingerprinting for Web clients[C]//Proceedings of the 6th Workshop on Security and Privacy in Smartphones and Mobile Devices. 2016: 93-102.
- [13] FAIZ-RHADEMI A, MOHAMMAD Z, KOMMINIST W. FPGuard: detection and prevention of browser fingerprinting[C]//Proceedings of the 29th Data and Applications Security and Privacy Conference. 2015: 293-308.
- [14] ROCHET F, KYRIAKOS E, OLIVIER P. SWAT: Seamless Web authentication technology[C]//Proceedings of the World Wide Web Conference. 2019: 1579-1589.
- [15] LAPERDRIX P, GILDAS A, NICK N. Morellian analysis for browsers: making web authentication stronger with canvas fingerprinting[C]//Proceedings of the 16th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. 2019: 43-66.
- [16] SecurAuth. Device/Browser fingerprinting-heuristic-based authentication[EB]. 2018.
- [17] Iovation. ClearKey[EB]. 2018.
- [18] ALACA F, VAN O. Device fingerprinting for augmenting web authentication: Classification and analysis of methods[C]//Proceedings of the 32nd Annual Conference on Computer Security Applications. 2016: 289-301.
- [19] VASTEL A, PIERRE L, ROMAIN R. FP-STALKER: tracking browser fingerprint evolutions[C]//Proceedings of the 39th IEEE Symposium on Security and Privacy. 2018: 1-14.
- [20] LI X, CUI X, WANG X. Constructing browser fingerprint tracking chain based on lstm model[J]. IEEE. 2018, 7: 13-18.
- [21] LI S, CAO Y. Who touched my browser fingerprint? a large-scale measurement study and classification of fingerprint dynam-

ics[C]//Proceedings of the ACM Internet Measurement Conference. 2020. 370-385.

- [22] ANTONIO E, FAJARDO A, MEDINA R. Tracking browser fingerprint using Rule Based Algorithm[C]//16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA). 2020: 225-229.

[作者简介]



高凡（1996-），女，山东烟台人，北京交通大学硕士生，主要研究方向为网络安全及浏览器检测。



王健（1975-），男，山东烟台人，博士，北京交通大学副教授、博士生导师，主要研究方向为密码应用及区块链、网络安全。



刘吉强（1973-），男，山东烟台人，博士，北京交通大学教授、博士生导师，主要研究方向为可信计算、隐私保护、云计算安全。