软件学报 ISSN 1000-9825, CODEN RUXUEW
*Journal of Software*, [doi: 10.13328/j.cnki.jos.006025]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

# 大型指纹库场景中加密视频识别方法*

吴 桦[1,2,3], 于振华[1], 程 光[1,2,3], 胡晓艳[1,2,3]

[1] (东南大学 网络空间安全学院,江苏 南京 211189)

[2] (东南大学 教育部计算机网络和信息集成重点实验室,江苏 南京 211189)

[3] (东南大学 江苏省计算机网络重点实验室,江苏 南京 211189)

通讯作者: 吴桦, E-mail: hwu@seu.edu.cn

**摘 要**: 加密视频识别是网络安全和网络管理领域亟待解决的问题,已有的方法是将视频的加密传输指纹与视频指纹库中的视频指纹进行匹配,从而识别出加密传输的视频.现有研究主要集中在匹配识别算法的研究上,但是没有专门针对待匹配数据源的研究,也缺少在大型视频指纹库里对这些算法的查准率和假阳率指标分析,由此造成现有成果的实用性不能保证.针对这一问题,首先分析使用安全传输层协议加密的应用数据单元(Application Data Unit,ADU)密文长度相对明文长度发生漂移的原因,首次将 HTTP 头部特征和 TLS 片段特征作为 ADU 长度复原的拟合特征,提出了一种对加密 ADU 指纹精准复原方法 HHTF,并将其应用于加密视频识别.基于真实 Facebook 视频模拟构建了二十万级的大型指纹库.从理论上推导并计算出,只需要已有方法十分之一的 ADU 数目,在该指纹库中视频识别准确率、查准率、查全率达到 100%,假阳率达到 0.在模拟大型视频指纹库中的实验结果与理论推导结果一致.HHTF 方法的应用使得在大规模视频指纹库场景中识别加密传输的视频成为可能,具有很强的实用性和应用价值.

**关键词**: 加密视频识别; 应用数据单元; 传输指纹; 大型视频指纹库 ; 安全传输层协议

**中图法分类号**: TP393

中文引用格式: 吴桦,于振华,程光,胡晓艳.大型指纹库场景中加密视频识别方法.软件学报. http://www.jos.org.cn/1000-9825/6025.htm

英文引用格式: Wu H, Yu ZH, Cheng G, Hu XY. Encrypted Video Recognition in Large Fingerprint Database. Ruan Jian Xue Bao/Journal of Software, (in Chinese). http://www.jos.org.cn/1000-9825/6025.htm

## Encrypted Video Recognition in Large-scale Fingerprint Database

WU Hua[1,2,3], YU Zhen-Hua[1], CHENG Guang[1,2,3],HU Xiao-Yan[1,2,3]

[1] (School of Cyber Science and Engineering, Southeast University, Nanjing 211189, China)

[2] (Key Laboratory of Computer Network and Information Integration of Ministry of Education of China, Southeast University,Nanjing 211189, China)

[3] (Key Laboratory of Computer Network Technology of Jiangsu Province, Southeast University, Nanjing 211189, China)

**Abstract**: Encrypted video identification is an urgent problem in the field of network security and network management. The existing methods are to match the video transmission fingerprint of encrypted video with the video fingerprint in the video fingerprint database. The existing research mainly focuses on the study of matching recognition algorithm, but there is no particular research on matching data sources, nor the analysis of precision and false positive rate in large-scale video fingerprint library. The resulting practicality of existing

* 基金项目: 国家重点研发计划项目课题(2017YFB0801703,2018YFB1800602); 教育部-中国移动科研基金(MCM20180506);国家自然科学基金项目(61602114);赛尔网络下一代互联网技术创新项目(NGIICS20190101,NGII20170406)

Foundation item: National Key R&D Program of China (2017YFB0801703, 2018YFB1800602), Ministry of Education-China Mobile Research Fund Project (MCM20180506), the National Natural Science Foundation of China (61602114),the CERNET Innovation Project(NGIICS20190101,NGII20170406).

收稿时间: 2019-10-07; 修改时间: 2020-01-06; 采用时间: 2020-02-28; jos 在线出版时间: 2021-05-20

methods cannot be guaranteed. In order to address this problem, this paper firstly analyses the reason why the length of the ciphertext of the application data unit (ADU) encrypted by TLS drifts relative to the length of the plaintext. For the first time, HTTP head feature and TLS fragment features are used as fitting features for ADU length restoration, then this paper proposes an accurate fingerprint restoration method HHTF for the encrypted ADU, and applies HHTF to the encrypted video recognition. A large fingerprint database of 200,000 videos was built based on the simulation of real Facebook videos. Theoretical derivation and calculation demonstrate that the accuracy, precision and recall rate can reach 100%, and the false positive rate is 0 requiring only one-tenth the number of ADUs of the existing method. The experimental results in simulating large-scale video fingerprint database are consistent with the theoretical calculations. The application of the HHTF method makes it possible to recognize encrypted transmitted video in large-scale video fingerprint library scenes, which is of great practicality and application value.

**Key words**: encrypted video identification；application data unit； transmission fingerprint；large-scale video fingerprint database；Transport Layer Security Protocol

互联网最初的设计功能已经远跟不上实际的需求,隐私保护和安全防护是互联网应用必须考虑的问题,利用 TLS（Transport Layer Security,TLS）协议实现数据的端到端加密传输是最通用的加密传输方法.由于数据重要程度不一样,有些应用只对用户登录数据加密传输,有些是对所有数据都加密传输,随着硬件成本的下降和人们安全意识的提高,大趋势是所有数据加密,这些措施为互联网应用提供了很好的安全防护.但是另一方面,加密流量比重的增加给网络安全和网络管理带来极大的挑战.

如何从加密的数据中抽取出网络安全防护和网络管理需要的信息已经成为国家安全部门网络管理中亟待解决的问题,既要保护普通网民的隐私,也要及时发现因特网中传递的危害国家和社会安全的信息,这需要能够在不解密信息的前提下精准识别特定的被加密信息.

目前对加密流量的分析主要分为两大类：应用类型识别和内容识别.对加密流量的应用类型识别开展的比较早,包括的范围也比较广,包括对加密流量的识别[1],对网络流量应用类型的识别[2-4],对恶意软件流量的识别[5-7],对加密视频播放模式的识别[8], 对加密视频服务平台识别[9,10], 对加密视频服务质量识别[10-14].这类研究都不涉及到用户信息的具体内容识别.

在网络安全和网络管理领域有较大需求且最具挑战性的是对加密应用内容的识别,包括对视频的识别和网站访问行为识别.根据 2018 年思科公司的全球互联网流量研究报告[15],互联网全球流量中超过 70%为视频流量,预计到 2022 年,视频流量的比例将增长到 80%,并且世界上 TOP 视频服务商都已经采用了加密视频传输技术.在这个背景下,由于视频传播容易,影响力广,对加密视频的识别已经成为亟待解决的问题.与加密视频内容识别同等迫切的需求还包括加密网站访问的识别[16-18].这两类应用的流量占据了互联网流量的绝大部分,随着加密传输的普及,对这两类应用内容的识别成为网络安全管理的挑战.

本文的研究围绕加密视频内容识别展开. 对加密视频内容的识别目标是通过数据传输特征获知被传输视频的内容标签,而不是对视频的画面内容进行分析,以下简称为加密视频识别.由于应用层信息被加密无法直接分析,侧信道是对加密数据分析的一种常见途径,现有加密视频识别研究的基本思路是从网络层和传输层协议头部信息中提取出应用数据单元（Application Data Unit,ADU）的特征.ADU 是应用层信息被传输的数据单元[19],在 HTTP 传输协议中每个 HTTP 请求的资源就是一个 ADU.这些 ADU 的数据量长度和传输顺序构成了应用层信息的指纹,观测者有可能从这些 ADU 的特征识别出应用层信息.

已有的加密视频识别研究[20-26]存在三个主要问题.第一个问题是现有研究的关注点都在识别算法的研究上,即如何将采集到的加密传输数据与视频指纹库进行匹配以识别热点视频.但是识别算法的输入信息——待匹配的加密传输数据与指纹库中的指纹,这两者如何构建,以及由于不同的构建方法带来的数据原始偏差都没有进行深入研究.当指纹库规模变大后,这些不确定性会极大影响着这些识别算法的结果.第二个问题是现有研究对算法结果的评价指标不全面, 通常只有查全率,少有假阳率的测试指标,特别是没有大型指纹库场景下查准率和假阳率的评估; 第三个问题是现有研究都是基于小型甚至是微型视频指纹库进行实验验证,实验结果不能反映这些算法应用于大型视频指纹库的可行性,也没有文献对方法应用于大型指纹库的可行性进行理论探讨.因此,即使已经有了一些初步的探索成果,在大型指纹库场景中的加密视频识别还是空白,这也是国家网

络安全建设中亟待解决的问题.

针对上述问题,本文对加密视频识别的关键问题进行研究,主要成果包括:

(1) 针对加密视频识别中的待匹配数据源进行研究,深入研究 TLS 数据加密传输中数据偏移的基本原理,首次将 HTTP 头部特征和 TLS 片段特征作为 ADU 长度复原的拟合特征,提出了一个将 ADU 加密传输长度精准还原出明文长度的方法 HHTF (Http head & TLS fragmentation),该方法适用于多个视频服务平台的 ADU 长度精确还原;

(2) 给出了使用 HHTF 方法复原 ADU 长度后,在大型视频指纹库中识别加密传输视频的方法,从理论上计算了加密视频识别方法的准确率、查准率、查全率和假阳率,并通过在大型测试数据库的实验验证了将 HHTF 方法应用于二十万级 Facebook 指纹库的加密视频识别效果, 只需要 3 个连续的 ADU 就可以达到准确率、查准率、查全率都为 100%, 假阳率为 0 的指标要求.

本文第 1 节介绍加密视频识别的基本方法和国内外相关工作,第 2 节给出了 ADU 长度精准复原方法 HHTF 及其适用范围,第 3 节给出了使用 HHTF 方法后在大型视频指纹库中进行加密视频识别的方法,从理论和实验两个方面给出了方法的准确率、查准率、查全率和假阳率评估结果, 根据评估结果推断出使用 HHTF 方法只需要 3 个连续 ADU 就可以在二十万级 Facebook 视频指纹库中识别出视频,最后给出本文的结论并展望未来工作.

## 1 相关工作与本文的研究内容

### 1.1 加密视频识别基本原理

目前国外主要的视频分享服务商如 YouTube、Netflix,主要社交网站如 Facebook,都对其提供的视频服务采用了加密传输.国内的视频服务商虽然在用户认证环节已经使用了加密传输,但是视频数据传输还是明文传输.因此对加密视频的识别研究都是围绕国外视频服务平台进行的.

因为无法利用加密视频应用层的内容特征,对加密视频的识别主要的可利用特征是 ADU 的长度特征和传输顺序.现有的加密视频分发平台都使用了 HTTP 自适应流媒体技术(HTTP adaptive streaming,HAS),如 MPEG 与 3GPP 提出的基于 HTTP 的动态自适应流媒体技术 DASH (dynamic adaptive streaming over HTTP) [27] ,以及苹果公司的 HLS (HTTP live streaming) [28]方案.为了使得视频能够在播放过程中进行自适应切换,这些技术都是将视频文件按照视频的等长播放时间切成一系列的视频 ADU,以便客户端根据传输环境选择下载不同分辨率的视频.这些被切片的视频 ADU 播放时长是固定的,由于视频内容的不同,按序切分的 ADU 数据长度不一样,这样的顺序和长度就构成了一个视频的明文指纹,在实际传输时,由于应用数据被 HTTP 协议和 TLS (Transport Layer Security) 协议封装,传输数据量要比明文数据量略大,构成传输指纹. 图 1 即为 Facebook 视频“Avenger4:Endgame”分辨率为 360P 的明文指纹和传输指纹.

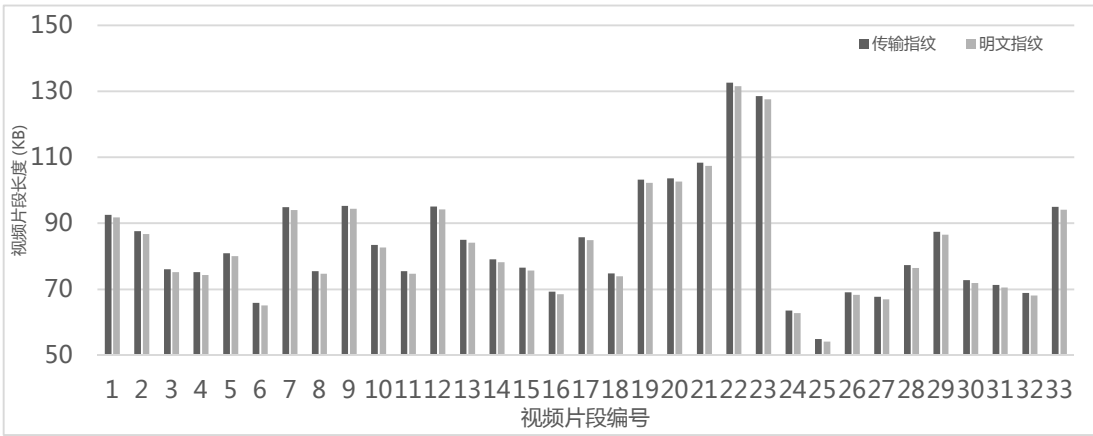



Fig.1 Video plaintext fingerprint and transmission fingerprint

图 1 视频明文指纹和传输指纹

这些视频片段在用户观看视频时是按序传输的.图 2 为使用 DASH 传输机制的示意图,客户端首先获取视频描述文件（Media Presentation Description,MPD）,解析后发起 HTTP 请求,每次请求的内容为 1 个视频 ADU,通过按序请求视频 ADU 可以在视频播放器完成播放.这些视频 ADU 在播放过程传输的 ADU 长度和传输顺序可以构成一个视频的传输指纹.

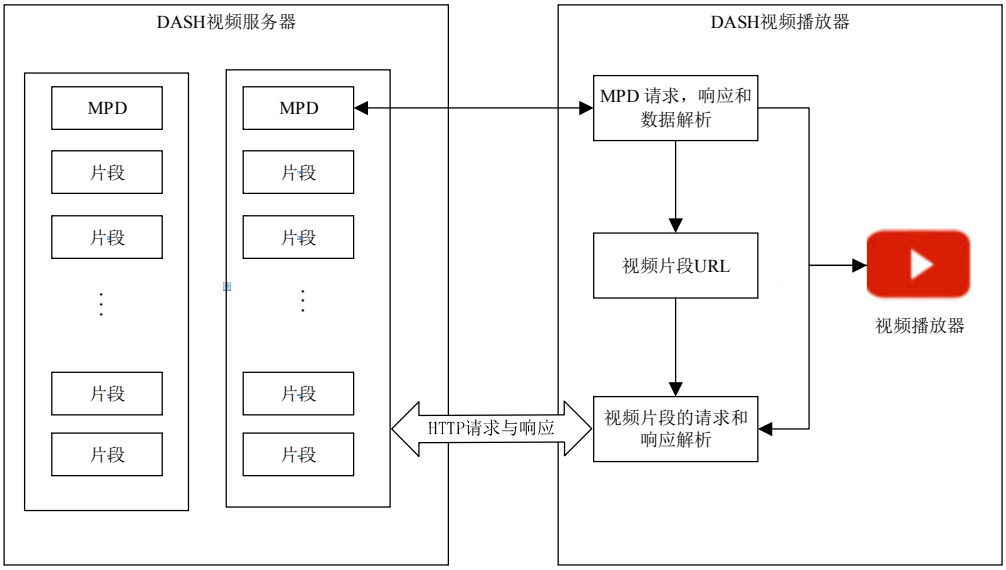



Fig.2 DASH Video Transmission

图 2 DASH 视频传输

加密视频的识别过程就是将已知的视频明文指纹与视频播放过程中的加密传输指纹 ADU 进行匹配,由于加密协议封装会导致 ADU 长度变化,不同内容的 ADU 加密后也可能具有相同的长度,这导致匹配结果是有误差的.因此需要给出匹配算法的评估指标,当匹配算法的评估指标在允许范围内,就可以认为加密传输的视频就是已知的视频.

但是,观看过程可能会发生分辨率自适应切换事件和用户手动改变播放进度事件,因为同一个视频不同分辨率的指纹是不一样的,切换分辨率就改变了明文指纹,同样改变播放进度也会导致视频的 ADU 不按顺序传输,对应明文指纹发生了变化,这些情况导致在现实中明文指纹和传输指纹很难全程匹配,这些情况下只有局部匹配是可能的.因此在实际进行匹配时,并不是将视频播放过程中的所有 ADU 进行匹配,而只是使用部分 ADU 与指纹库匹配,而且需要匹配的 ADU 数量越少越好.

在数据被加密传输的背景下,已有研究中获得 ADU 的长度特征都是利用了 HTTP1.1 流水线模式传输特点[19].在使用 HTTP1.1 流水线模式的 TCP 连接中,服务器响应给客户端的 ADU 是按照客户端的请求顺序发送的,对同一个请求的响应数据包序列其 TCP 头部的响应序列号是一样的,通过分析 TCP 报头信息,将属于同 1 个 ADU 的响应数据包负载长度进行加总,就可以得到 1 个 ADU 的 1 次加密传输的数据长度.自适应流媒体传输过程中 1 个视频 ADU 就是视频的一个片段.统计视频播放过程中的所有 ADU 加密传输的长度和顺序,就可以得到这次播放的视频传输指纹.

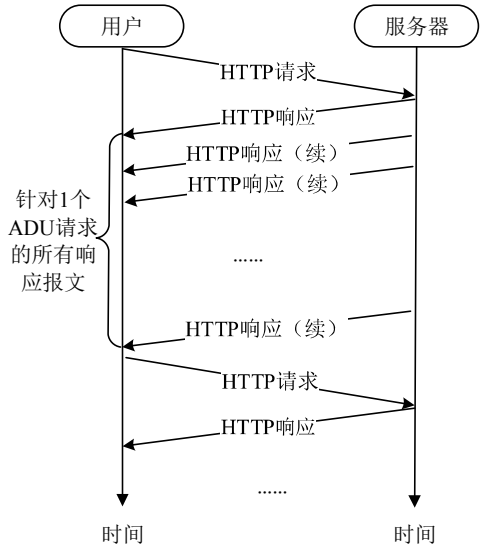



Fig.3 ADU Request and Response in HTTP 1.1

图 3 HTTP1.1 协议中 ADU 请求与响应

有了视频的明文指纹和传输指纹以后,将视频传输指纹与视频指纹库中明文指纹进行匹配,如果两者匹配成功,就可以识别出用户播放的视频内容.因此我们需要明确指纹库的构建方式.

在加密视频识别领域中已有文献对指纹库和指纹并没有统一的定义,为了明确本文陈述的内容,本文给出如下的名词定义:

定义 1. 明文指纹库. 用视频明文信息构建的指纹库.

定义 2. 明文指纹. 明文指纹库中的视频指纹.

定义 3. 密文指纹库. 用视频密文传输实例构建的指纹库.

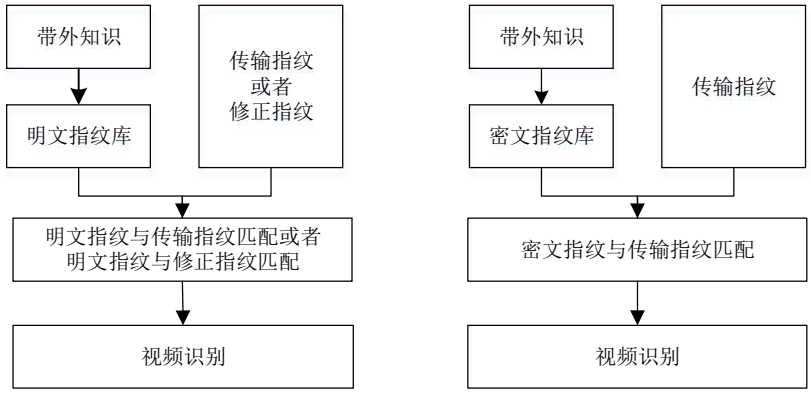
定义 4. 密文指纹. 密文指纹库中的视频指纹.

定义 5. 传输指纹. 视频 ADU 被加密传输时,从传输密文的侧信道提取的长度指纹.

定义 6. 修正指纹. 使用视频明文信息构建的指纹库识别时,为了使得传输指纹更接近明文指纹,对传输指纹进行修正后的指纹.

现有的识别方法在构建视频指纹库时使用了两类方法分别构建明文指纹库和密文指纹库.第一类方法是通过带外的方法获得视频明文信息,如中间人代理获得视频描述文件,这些描述文件是服务器提供给播放器的对每个视频 ADU 的描述,是对片段明文属性的描述,可以用来构建明文指纹库；第二类方法是直接在终端播放特定视频,同时中间节点采集对应的传输数据,将终端记录的视频名称和同时采集到的加密传输数据构成一个传输实例,将视频名称及播放时加密数据的传输特征存储到数据库中构建指纹库,这个指纹库里存储的是视频传输指纹,是对一次加密传输实例的描述,因此为密文指纹库.

基于上述定义和两类指纹库的构建方法,现有加密视频识别的基本方法分为两大类,如图 4 所示.




（a）使用明文指纹库进行视频识别　　(b) 使用密文指纹库进行视频识别

Fig.4 Encrypted video identification method

图 4 加密视频识别方法

如图 4 所示,根据指纹库构建方式的不同,加密视频识别方法分为两类.图 4(a)是使用视频明文信息构建的指纹库,利用带外知识为视频指纹打上内容标签,从侧信道提取的传输指纹进行修正后与明文指纹进行匹配,基于匹配结果识别视频.这类识别方法也包括对传输指纹不进行任何修正就将其与明文指纹匹配的方法;图 4(b)中使用视频密文传输实例构建指纹库,也是使用带外知识为指纹打上内容标签,从侧信道提取的传输指纹与密文指纹进行匹配,基于匹配结果确定是否识别出视频.

### 1.2 评估测度

为了对加密视频识别的效果进行评价,需要选择合适的性能指标.加密视频识别属于二分类任务,我们已知对二分类问题的预测,可以得到四种结果,分别为 *TN*（true positive）、*FP*（false positive）、*FN*（false negative）、*TP*（true positive）.在加密视频识别算法评价中,使用准确率（Accuracy）,查准率（Precision）,查全率（Recall）,假阳率（False Positive Rate）可以全面评价算法的有效性.计算公式分别为:

准确率的公式为: $A = \frac{TP + TN}{TP + TN + FP + FN}$, (1)

查准率的公式为: $P = \frac{TP}{TP + FP}$, (2)

查全率的公式为: $R = \frac{TP}{TP + FN}$, (3)

假阳率的公式为: $FPR = \frac{FP}{FP + TN}$, (4)

准确率,查准率,查全率,假阳率必须联合使用以全面评测算法的可用性,如果对算法结果只评测其中个别指标,即使个别指标结果很好,其他关键指标没有评测,算法的有效性也无法保证.

### 1.3 相关工作

本节首先对已有研究成果结合图 4 中指纹库的不同构建方法分类阐述,然后讨论这两种方法构建的指纹库的区别,从而确定本文的指纹库构建方法.

图 4(a)中使用明文信息构建明文指纹库是最直接的方法.首先分析使用明文指纹库的相关文献.

Reed 等[20]开发了一个能够识别加密 Netflix 视频的系统.该系统使用中间人代理获得的视频描述信息构建明文指纹库,对加密视频识别时,通过 adudump[19]提取的加密 ADU 特征构建视频传输指纹.但是通过 adudump

提取的传输指纹与明文指纹库中的明文指纹长度上存在偏移.Reed 等考虑到这个问题,指出 HTTP 头部和 TLS 协议开销会对数据造成影响,通过将匹配窗口放大到 30 个 ADU,以及对 ADU 特征进行一些修正,将这个影响尽量降低.该文献在一个包括 330364 个 Netflix 视频指纹库中做了 200 次识别测试,测试结果为 199 次正确识别出视频,即该方法查全率是 99.5%,但是该文献没有给出其他的评测指标.Reed 等的另一篇论文在 802.11 无线网络中识别加密的 Netflix 视频流[21] ,但是该文献的测试指纹库只有不到 100 个视频,进行了 25 次识别全都识别出视频,因此查全率为 100%,除此之外没有给出其他评测指标.这篇论文数据库规模太小,该文也指出该方法的误判率随着指纹库规模增大会增大,无法应用到实际场景.这两篇论文都要求加密视频数据采集达到 30 个 ADU 才能进行匹配,即采集 30 个连续的 ADU,并且在此期间没有分辨率切换及人工跳转才能用于视频识别.

Stikkelorum 等[22]使用有限状态机进行视频识别,使用文献[20]中的修正方法对 ADU 特征进行修正,修正后的视频传输指纹与明文指纹库进行匹配.这篇文献的指纹库只包括 20 个 YouTube 视频,测试结果也只是在这 20 个视频的指纹库里依次识别 5 个视频并只给出查全率,从指纹库的规模和算法的评估结果看,该文献成果不具有实用性.

图 4(b)中使用加密传输的信息构建密文指纹库也是常用的指纹库构建方法,通常用于无法获得明文指纹的场景中.

Gu 等人[23,24]提出一种从侧信道识别视频的方法,指纹数据来源于传输过程中的吞吐量变化,因此属于密文指纹,传输指纹是从视频播放时的数据侧信道中提取的,因此这个方法本质上是将密文指纹与传输指纹进行匹配.测试时指纹库有 200 个视频,查全率为 90%,并没有给出假阳率,该方法要求采集可播放 3 分钟的密文数据,对应 Facebook 数据为 90 个 ADU.同时,该方法的测试数据是实验网采集,而现实场景中的背景流会干扰该算法假设的视频流固定传输模式,该文献的结论也指出该方法中无法识别出 ADU,因此尚无法应用在大规模指纹库场景中.

文献[25]提出了一种识别 Netflix 交互视频用户动作的方法,指纹库是通过用户实际操作的动作结合动作发生时抓取的密文构建,属于密文指纹库,传输指纹来自于客户端 TLS 记录协议长度,使用的是密文指纹与传输指纹进行匹配的方法.该文献针对一个交互视频中的 10 个选择点构建指纹库,测评结果是该算法达到 96%的查全率.由于指纹库太小,没有给出假阳率,该成果也无法推广到大规模指纹库.

文献[26]认为一个视频的指纹是固定的,因此多次下载模式是固定的.但是该文献并没有使用指纹库,该方法对一个视频的播放模式进行机器学习训练分类器,对不同的视频需要训练不同的分类器,再提取监听到的视频播放特征进行分类识别.这篇文献方法需要对每个视频训练一个分类器,代价太高,而且一个重要的假设是同一个视频在网络上的传输模式是固定,这个假设在广域网上并不成立,主干网上单个应用流得到的可用带宽是波动的,导致每次的传输模式并不是固定的,该文献对数据采集环境要求较高,因此并不适合在大规模网络上应用.

总体看来,文献[23-26]的密文指纹构建密文指纹库的方法都面临两个问题:(1)密文指纹库存在指纹库内容不确定,方法各不相同导致结果无法具有通用性;(2)每次对 ADU 加密后的长度并不能保证不变,引起不确定性的因素包括 HTTP 头部信息每次传输都有可能会变化,每次传输时服务器的性能状态不一样也会导致 TLS 片段数目不一样,相应会添加不确定数目的 TLS 片段头部[29],这些不确定因素造成一个 ADU 的密文长度会有多种,使用不确定的长度构建指纹库会为后续匹配带来误差.为了避免使用不确定性信息构建指纹库,本文研究使用明文指纹构建视频指纹库的方法.

由现有文献分析可见,无论使用明文指纹库还是密文指纹库,现有文献在视频内容识别领域内所做的研究都处于初始的探索阶段,存在的问题也比较相似:(1)主要研究点集中在各种匹配算法的优化研究上,但是没有文献深入研究匹配算法的输入数据是否合理可信,待匹配的信息来源比较混乱,这必然降低了这些方法的通用性及其评测结果的准确性.(2)对算法结果的评测指标不全面,这一问题在已有文献中体现为对算法的评测指标主要为识别的查全率,而假阳率只在个别使用小型指纹库测试的文献中被提到,但是指纹库很小的情况下假阳率是没有参考价值的.(3)测试指纹库普遍比较小,评测结论不一定适用于大型指纹库.这些问题说明这些

加密视频识别研究成果只是初步的尝试,尚无法解决在真实场景中的加密视频识别问题,也说明了在加密流量比例逐步提升的现实场景下网络安全和网络管理面临的困难.

### 1.4 本文的研究内容

本文针对加密流量识别研究中的关键问题展开工作,研究加密视频传输指纹的精准还原方法及其在加密视频识别方面的应用价值.这两个研究内容的关系如图 5 所示:

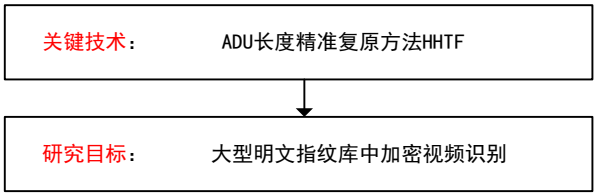



Fig. 5 The key research point of this paper

图 5 本文的关键研究点

ADU 长度精准复原方法 HHTF 可以从加密传输的 ADU 复原出 ADU 明文长度,这是本文的关键技术创新点,这一技术大大提高了加密视频识别结果的准确率、查准率、查全率,降低了假阳率.使用本文的方法后进行加密视频识别能够实现大型明文指纹库场景中加密视频的准确识别.

## 2 ADU 长度精准复原方法 HHTF

对单个 ADU 的长度精准复原是 ADU 匹配的前提,本节给出了对单个 ADU 长度进行精准复原的方法,该方法的关键点在于特征的提取考虑了 HTTP 头部和 TLS 片段这两个关键因素,因此在下文中简写为 HHTF(Http head & TLS fragmentation)方法.

本节首先给出复原方法的总体架构,然后着重阐述了 TLS 加密数据长度偏移的基本原理,基于这个基本原理给出了特征提取的方法,使用提取的特征进行模型拟合得到 HHTF 修正方法的参数,并讨论了 HHTF 方法的适用性.

### 2.1 加密应用数据单元长度精准复原方法架构

图 6 为本文提出的加密应用数据单元长度精准复原方法架构图:

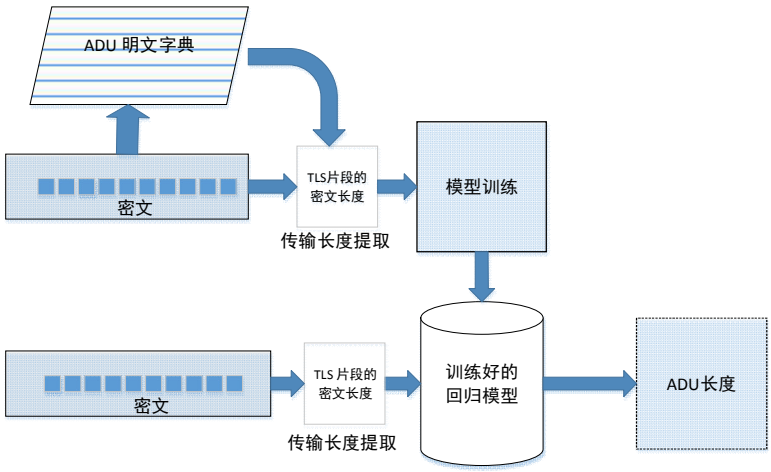



Fig.6 Architecture of the accurate restoration method for encrypted ADU length

图 6 应用数据单元长度精准复原方法架构

首先通过代理等带外方式采集应用的明文数据信息,并提取其长度信息构成 ADU 明文字典.需要指出,此处的 ADU 明文字典与图 4 的视频明文指纹是不同的.此处研究的是单个 ADU 长度的复原方法,明文字典里存储的是应用层单个 ADU 的长度,只有数据量特征,而图 4 的视频识别应用中,指纹库中的视频指纹包括一系列 ADU 的长度及其传输的时间顺序特征.

通过明文字典对训练数据打上长度标签,并提取密文传输时的传输长度和相关特征,再通过机器学习得到对 ADU 长度精准复原的回归模型.对 ADU 长度进行修正时,提取 ADU 加密数据的传输长度和相关特征,使用训练好的回归模型进行计算,就可以精准复原出该 ADU 的明文长度.

### 2.2 数据集

由于尚无公开的视频明文与密文对应的数据集,本文采集了 Facebook 的数据集,采用了如下的方法.

针对明文字典的构建,我们通过对 DASH 视频传输时的 MPD 文件解析,获得明文的准确信息.MPD 文件是 DASH 模式中视频 ADU 的元文件,包含了视频 ADU 信息以及视频 ADU 资源地址信息.使用 DASH 模式传输视频时,在每次播放的开始以及分辨率切换时,会传输该视频对应分辨率的 MPD 文件.通过对 MPD 文件的解析,我们可以获得这些视频片段(即视频 ADU)的明文特征,包括 ADU 的数据量长度.MPD 文件也是加密传输的,为了获得 MPD 文件的内容,将移动终端通过 PC 提供的热点接入网络,在 PC 上运行中间人代理.在移动终端点播 Facebook 不同视频,并手动切换不同分辨率,就可以通过中间人代理获得 MPD 文件的明文,进而对 MPD 文件进行解析,获得视频 ADU 的描述信息.这些信息可以用来构造 ADU 明文字典.

为了获得密文传输实例,移动终端使用 PC 上的热点,启动接入热点上的 Wireshark,在移动终端上点播视频,视频播放的时候就可以在 PC 上抓取密文数据.实验数据采集过程中严格顺序播放视频,并在实验后释放应用缓存空间,以保证每次播放时都是全数据传输,这样可以依次正确提取视频 ADU 的传输长度.由于接入网速的限制,采集的这些传输指纹样本主要由 144P,240P,360P 这三种不同的分辨率组成,分析这些数据获得可用的视频传输 ADU 密文 14551 个.

### 2.3 单个ADU传输特征提取

#### 2.3.1 TLS 加密数据传输长度偏移基本原理分析

加密 ADU 的传输长度与其对应的明文长度进行匹配时,传输长度越接近明文长度,则匹配越准确.但是在加密传输的情况下,我们只能得到所有加密数据包载荷长度之和 $Payload\_S_c$,由于网络协议添加了多种信息头部, $Payload\_S_c$ 相对明文长度有了偏移,必须将 $Payload\_S_c$ 修正成接近明文长度的值,再与明文长度匹配.本节分析 TLS 协议加密后数据传输长度发生偏移的原因,这是特征提取的关键点.

在目前所有相关研究中,ADU 的数据长度特征提取都是直接使用文献[19]提供的工具或者开发的类似工具,将对同一个 HTTP 请求的响应数据包应用层载荷长度之和视为一个 ADU 的长度.但是实际情况并非如此,如图 3 所示,应用层的 ADU 需要经过 HTTP 协议、TLS 协议、TCP 协议封装后才能成为 TCP 数据包.TLS 加密数据通过 TCP 协议传输时,只能获得 TCP 头部信息和 IP 头部信息,TCP 的载荷大部分是加密的.为了分析数据长度发生的变化,首先需要明确 ADU 转换为 TCP 数据包的过程中发生的信息变化.

自适应流媒体 MPEG-DASH 或者 HLS 模式传输流媒体视频,都是使用的 HTTP 应用协议,因此如图 7 所示,应用层 ADU 首先由 HTTP 协议封装,随后 ADU 和 HTTP 头部合并后通过接口调用被 TLS 协议处理,首先会被分片,然后可能会被压缩、添加 MAC(Message Authentication Codes)值,随后加密成为一系列的 TLS 片段.这些 TLS 片段都会有一个 TLS 头部结构,含有数据类型、版本号和长度信息等信息.这些 TLS 片段成为 TCP 传输协议的载荷.

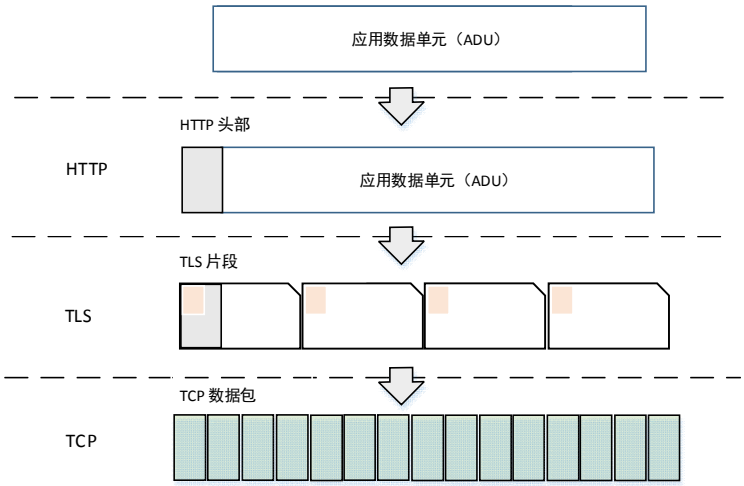



Fig.7 The process of encapsulating an application data unit to a series of encrypted TCP packets

图 7 将一个应用数据单元封装为一系列加密 TCP 数据包的过程

从图 7 可以看出,TCP 数据包载荷长度之和与 ADU 的数据长度必然存在偏移,这些偏移包括增加了 HTTP 头部信息、TLS 头部信息.由于 TLS 协议将 HTTP 头部和应用数据单元切分为一些 TLS 片段后加密.每个 TLS 片段头部都会增加 TLS 头部信息,片段数目越多增加的头部信息越多,因此 TLS 片段的数目也是影响 ADU 长度偏移的关键因素.

HTTP 头部信息在 TLS 片段中有两种分布方式,如图 8 所示.第一种是 HTTP 头部与加密数据被混在一个 TLS 片段中,第二种是 HTTP 头部单独成为一个 TLS 片段.通过对 Facebook 和 YouTube 数据的分析发现,Facebook 超过 85%的样本,YouTube 的全部样本都是按照图 8 中的 TLS 片段数据分布 2 所示分布的.这是因为视频服务器响应时,HTTP 头部信息是由服务器直接产生的,而视频数据是从硬盘中读出的,这两者到达缓冲区的速度不一样,从而导致先到达的 HTTP 头部作为一个单独的 TLS 片段,而且这个片段长度的分布具有明显的区间范围,如 Facebook 平台这个 TLS 片段长度会分布在[400Byte,700Byte]内.图 8 中的 TLS 片段数据分布方式 1 实际中占比很少,本文的策略是视为不可用过滤掉.在 2.2 节采集的数据集中,只有 12%的密文数据是属于这种情况的.在实际的视频应用中,出现分布 1 的概率会小很多,因此过滤这样的数据并不影响本文方法的适用性.

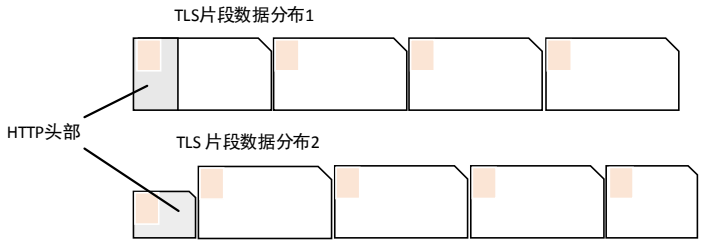



Fig.8 Position of HTTP header in TLS fragmentation.

图 8 HTTP 头部在 TLS 片段中的位置

### 2.3.2 特征值提取

ADU 长度精准复原的关键点在于将上述造成 TLS 加密数据传输长度偏移的因素加入数据特征的选择.本文选用三个特征值：*Payload_S*$_c$、*HTTPhead_L* 和 $N_{TLS}$.这三个特征的具体含义如图 9 所示：

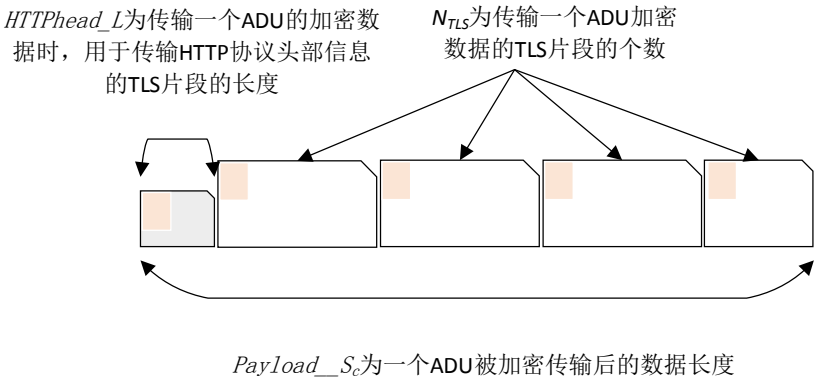



Fig. 9 The meaning of the three features
图 9 三个特征值的含义

*Payload_*$S_c$ 特征的提取方法采用的是类似于文献[27]中的方法,将在传输层获得的应用层载荷之和作为 *Payload_*$S_c$.

对 *HTTPhead_L* 和 $N_{TLS}$ 特征的选取是以图 7 和图 8 所示的原理为依据.因为想复原 ADU 明文的数据长度,必须在加密数据长度中减去 HTTP 头部的数据长度和 TLS 头部的数据长度.因此 HTTP 头部对应的密文长度,以及 TLS 片段个数必然为主要特征.根据文献[29],TLS 片段的长度最大为 16KB,再加上 TLS 片段头部信息,总长度通常大于 TCP 数据包的最大长度 MSS（Maximum Segment Size）,因此 TLS 片段会被分割在若干 TCP 数据包中发出,并且在两个 TLS 片段的交界处,分别属于两个 TLS 片段的数据会合成一个 TCP 数据包发出.从密文中提取 $N_{TLS}$ 就需要进行反向操作,如图 10 所示,将一个应用层数据单元的所有 TCP 数据包重新拼装为真实的 TLS 片段,才能得到对应的 TLS 片段个数.

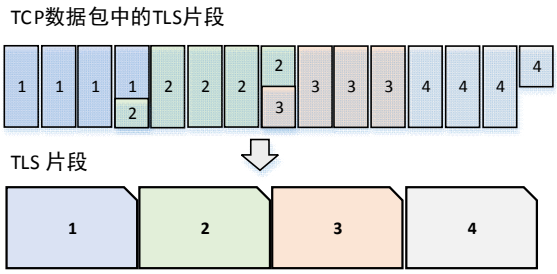



Fig. 10 Combining TLS fragmentation from TCP packets
图 10 从 TCP 数据包中组合出 TLS 片段

TLS 片段个数无法直接从 TCP 和 IP 的报头得到,需要结合 TLS 头部信息的解析得到.在 TLS 片段的头部所包含的 TLSPlaintext 结构中,包含了该 TLS 片段的长度信息,这些信息并不是加密的.因此可以解析 TCP 数据包载荷中的 TLS 片段头部信息,得到每个 TLS 片段的长度信息,再根据每个 TCP 载荷的实际长度信息,将 TCP 数据包合并或者拆分到各 TLS 片段中,从而组合出 TLS 片段,对组合出的 TLS 片段,根据上文分析的结论,如果第一个 TLS 片段长度在 400B 到 700B 之间,这个 TLS 片段包含的数据是 HTTP 协议的头部信息,将其长度提取为 *HTTPhead_L*.对剩下的 TLS 片段计算片段的个数,就获得了 $N_{TLS}$.

对每个 ADU 经过加密传输后得到的加密数据提取 *Payload_*$S_c$、*HTTPhead_L* 和 $N_{TLS}$这三个特征,结合之

前对这些 ADU 做的明文标记,就构成了训练集和测试集.

### 2.3.3 特征值提取中需要解决的关键问题

2.3.2给出的是特征值提取的基本原理和方法.前提条件是能够得到ADU的所有数据包,虽然文献[19]以及相关的研究都是利用了图 3 所示的基本原理,但是在处理实际的传输数据时,实际情况复杂很多.主要表现在:

(1) 数据传输必然存在丢包、重传、乱序的现象;

(2) 数据采集的时候可能由于采集系统的性能出现漏采集的现象;

(3) 客户端接收服务器发送的 ADU 的时候,可能由于网络状况的恶化中断已有的传输,然后客户端重新请求 TCP 连接,并发出续传请求,中断后续传的起点会根据不同情况有所不同,这导致 1 个 ADU 的数据可能来自 1 个 TCP 连接或者多个 TCP 连接;

(4) 当发生分辨率自适应切换的时候,在切换处会出现多余的 ADU;

(5) 用户在播放过程中的暂停、回放、快进等操作导致的数据复杂化.

在数据被加密的背景下上述这些情况需要能够被识别并进一步处理,由于这部分技术细节的解决过程颇为复杂,限于篇幅有限,以及这部分数据预处理内容更偏向于工程实现,具体细节不在本文中展开.

这些由于网络传输的复杂性导致的问题在已有的相关文献中都没有被提及,如果直接忽视这些细节是无法准确得到本文提出的三个特征值的.本文在数据处理过程中充分考虑了网络传输复杂性带来的问题,这是 HHTF 能精准复原明文长度的技术支撑.

## 2.4 回归模型拟合结果

根据图 7 给出的 TLS 传输长度漂移原理,计算 ADU 长度精确复原值 $ADU\_R$ 的公式为:

$$ADU\_R = Payload\_S_c - HTTPhead\_L - N_{TLS} \times \theta \tag{5}$$

$ADU\_R$ 为将加密数据长度复原后获得的长度,$\theta$ 为数据中每个 TLS 片段增加的信息的长度.$\theta$ 的取值与加密数据传输使用的 TLS 协议版本以及加密套件相关,为准确起见,对不同的 TLS 协议版本或加密套件需要提取特征后进行模型拟合,得到 $\theta$ 值.

根据 2.3 节的特征提取方法,对 Facebook 样本的 ADU 传输数据提取特征,并使用带外方式打上明文长度标签,进行模型训练后回归模型为:

$$ADU\_R = Payload\_S_c - HTTPhead\_L - N_{TLS} \times 29 \tag{6}$$

即 Facebook 数据拟合后 $\theta$=29,说明 Facebook 对视频数据进行 TLS 加密时,每个 TLS 加密片段增加 29 字节的头部信息.

对数据集中符合要求的12739个ADU传输指纹使用公式(6)计算了 $ADU\_R$,和明文指纹 $ADU\_F$ 比较,12739个计算结果和明文数据完全吻合,计算结果表明,HHTF 方法得到的修正值是一个确定性变量,而不是随机变量,由于 HHTF 方法修正后得到的修正长度等于明文长度,HHTF 可以精准复原 ADU 长度.

HHTF 可以精准复原长度的原因有两个:1) 本模型是根据加密流程的基本原理推理的,特征选择包括了所有影响长度的因素;2) 少数无法获得 HTTP 头部加密长度准确值的情况,即符合图 7 中的 TLS 片段数据分布 1 的数据样本不参与训练,也不参与测试.

理论上说,在 TLS 协议中的压缩、填充也会影响数据长度,但是在实际监测中发现,对现有视频数据来说,视频明文本身就是压缩的,二次压缩没有效果,因此都没有在 TLS 里实现压缩.有关数据填充问题,本文的研究过程中也发现,TLS1.0 会有数据填充,而现在普遍使用的 TLS1.2 传输,经过对 YouTube 和 Facebook 数据的分析调查,在传输视频数据时都没有填充,因此本文提取的特征值对 TLS1.2 加密传输的视频已经足够,可以得到 ADU 长度精准复原值.

HHTF 方法之所以能高度准确复原 ADU 的数据长度,是因为特征的提取考虑了 HTTP 头部和 TLS 片段这两个关键因素.下面从视频服务平台和终端两方面讨论其适用性.

### 2.5 HHTF方法的适用性

除了 Facebook 的视频片段,我们同时测试了 YouTube DASH 视频片段,由于 YouTube 默认情况使用 QUIC 协议传输视频,在接入路由器上关闭 UDP 协议的 443 端口后,YouTube 就恢复使用 HTTPS.用同样的方法采集了测试数据集.YouTube 每个 ADU 的可播放时长为 10 秒,本文采集了 376 个片段的传输指纹,构建了对应明文指纹库,同样进行了模型训练,得到的模型与公式(6)一样.使用公式(6)对传输指纹进行修正,再与明文指纹比较,376 个片段的修正结果与明文指纹库的长度完全吻合.所有的传输指纹都可以还原到与明文指纹精准匹配.由此可见本方法同样适用 YouTube 视频 ADU 传输指纹的还原.

此外,对 YouTube 的实验样本分析结果发现,YouTube 样本全部符合图 8 中的 TLS 片段数据分布 2,也就是 HHTF 方法完全可以适用于 YouTube 视频 ADU.由于 Netflix 需要当地移动接入的移动终端才能播放,本文没能采集数据进行验证.但是从加密视频服务器平台的覆盖面上看,Facebook 和 YouTube 的测试结果已经可以说明 HHTF 方法的适用性.

本文实验数据采集使用了三星 Note5、华为畅享 5、三星 s5 和三星 s6 edge 四款手机,在所有 4 个测试手机上,Facebook APP 使用 TLS1.2 协议时都选了加密套件“ TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xC02B) ” ,而 YouTube 的 APP 使用 TLS1.2 协议都选用了加密套件“TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xC02F)”.虽然加密套件不同,但是本方法都适用.

由此可见 HHTF 方法不仅适用不同的视频分发网站,对移动终端也有较广的适用性.

## 3 大型明文指纹库中加密视频识别

### 3.1 大型明文指纹库的构建

为了评估 HHTF 方法应用的效果,必须构建大型的视频指纹库.视频指纹库中存放了视频的 ADU 长度及其播放顺序,这些信息构成了视频的指纹.

由于获得 Facebook 真实的大型视频指纹库在现有条件下难以办到,本文基于统计学的基本原理构造大型模拟视频指纹库,只要样本具有独立性和代表性,在样本容量足够大的情况下,可以从样本统计量推断总体参数,据此可以模拟构建大型 Facebook 视频指纹库.

首先需要获得真实的视频及视频 ADU 分布.为了能从样本统计量准确推断出总体统计量,样本的选择必须具有独立性和代表性.通过代理采集了真实的 Facebook 视频 277 个,视频的种类包括影视、体育、游戏、音乐和综艺五大类,五类视频采集的个数依次为 98 个、65 个、30 个、42 个和 42 个.视频的播放时长包括[1min,2min]、[2min,5min]、[5min,15min]、[15min,120min] 4 个时间长度区间.277 个视频的 ADU 片段数目共为 77802 个.同时也采集了播放这些视频的密文数据实例用以进行视频识别实验.图 11 是这 77802 个 ADU 长度的概率密度函数（Probability density function,PDF）.

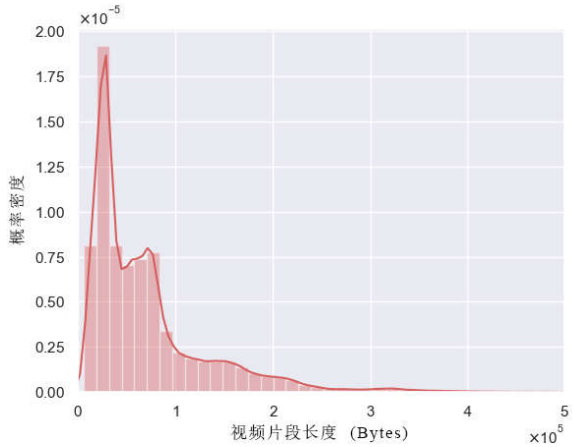



Fig.11 PDF of Facebook ADU length

图 11 Facebook 样本视频 ADU 长度概率密度

对于视频识别测试来说,277 个视频构成的指纹库远远不够.虽然我们无法得到 Facebook 的总体视频片段长度分布,但是已经采集的 277 个视频包含了 77802 个视频片段,因为视频片段的样本容量足够大,所以样本的分布逼近总体的分布.因此我们可以基于图 11 所示 77802 个 ADU 长度 PDF 构建一个模拟的大型视频指纹库.

大型视频指纹库的构成分为三部分:(1) 真实采集的 277 个视频;(2) 以每个真实视频为基础分别模拟出 200 个模拟视频构成了 55400 个模拟视频.这些模拟视频和真实视频 ADU 个数一样,ADU 长度随机分布在其对应的真实视频 ADU 长度 [0.9, 1.1]倍区间内;(3) 模拟产生了 150000 个视频,这些模拟视频 ADU 个数随机分布在[30, 930]范围内,ADU 长度按照图 11 中的概率密度函数产生.最终产生的模拟指纹库中含有 205677 个视频,87523677 个 ADU,平均每个视频 426 个 ADU,ADU 长度均值为 70KB.

这样产生的模拟数据库有三个特点:(1) 保证真实的视频包含在其中;(2) 包含了较多与真实视频指纹非常相近的视频指纹,因此可以用以检验是否会将指纹接近的视频混淆,在较为苛刻的情况下进行测试;(3) 视频的 ADU 长度是按照真实视频 ADU 长度的概率密度函数产生,因此整个模拟视频指纹库的 ADU 长度分布与真实的 Facebook 视频是一致的.

本文对视频的匹配方法是基于视频 ADU 长度和顺序进行的,模拟指纹库的 ADU 长度分布基于统计理论原理接近真实指纹库,完全可以用于对本文的算法进行验证.

### 3.2 ADU匹配算法与匹配概率

单个 ADU 是构成视频指纹的基本元素,也是进行加密视频识别的基础.本节给出将 HHTF 方法应用于单个 ADU 匹配时的方法和匹配概率,并给出对比的 Reed 方法应用后的匹配算法和匹配概率.

#### 3.2.1 HHTF 方法应用于单个 ADU 匹配算法及其匹配概率

根据第 2 节的结果,对符合要求的加密 ADU,HHTF 方法得到的长度复原值 *ADU_R* 与 ADU 明文的长度 *ADU_F* 是一致的,即获得的是确定性变量,所以在识别时使用的方法是 *ADU_R* 等于 *ADU_F* 视之为匹配.

匹配概率决定着匹配结果的准确性,匹配概率与数据库大小有密切的关系,本节使用 3.1 节构建的大型指纹库进行分析.

HHTF 方法进行修正后得到的为确定性变量,假设修正后得到长度为 $x$,事件 $A$ 为任意明文指纹长度和修正值 $x$ 匹配,事件 $A$ 的概率记为 $P(A)$, 使用 HHTF 方法修正后发生事件 $A$ 的概率记为:

$$P(A)=\int_{C1}^{C2} f(x)dx \tag{7}$$

公式(7)中的 *f(x)*为图 11 中的概率密度函数,*c1* 和 *c2* 是匹配的上下界,因为 HHTF 方法修正得到的是确定性变量,所以 $P_{HHTF}(A) \approx f(x)$,为了简化计算 *f*(*x*)可以使用 ADU 长度均值 $x_0$ 在总体中的概率 $f(x_0)$ 来估算,得到:$P_{HHTF}(A) \approx f(x_0)$.

根据 3.1 中模拟的测试指纹库的构建参数,可以得到 $f(x_0)=7.9\times10^{-6}$,即 $P_{HHTF}(A) \approx 7.9\times10^{-6}$.

### 3.2.2 Reed 方法应用于单个 ADU 匹配算法及其匹配概率

现有的对加密视频识别论文主要关注点在视频匹配算法的设计上,大部分都忽视了加密数据经过传输协议和加密协议封装后数据长度的不确定性,这是导致现有文献的成果无法真正应用到真实网络中的根本原因.目前对这问题提出解决方法的有文献[20-23],其中文献[20]与本文的方法一样使用的是明文指纹库,在进行匹配前对传输密文指纹做了修正.文献[21]发表于[20]之前,虽然有指纹修正,但只是简单等比扩大匹配范围,文献[22]则明确指出其参考了文献[20-21]的方法和参数,因此本文与文献[20]进行对比分析,以下对使用文献[20]的方法进行修正后匹配的方法称为 Reed 方法.

与 HHTF 对比的 Reed 方法中,文献[20]并没有对单个 ADU 匹配的方法及匹配分析,本节基于文献[20]的修正原理对单个 ADU 进行了修正,并给出了修正结果应用于单个 ADU 匹配的方法.

文献[20]中指出了直接使用密文传输指纹匹配明文指纹会产生偏差的原因:HTTP 头部对每个视频 ADU 增加大约 520 个字节;TLS 头部对视频 ADU 和 HTTP 头部的组合增加大约 0.18%的载荷.文献[20]在匹配时针对这两个偏差对传输指纹进行了边界修正:

$$Min = \frac{Total\_Received}{1.0019} - (30\times525) \tag{8}$$

$$Max = \frac{Total\_Received}{1.0017} - (30\times515) \tag{9}$$

Reed 方法要求连续采集到 30 个 ADU 才能进行视频匹配,因此 *Min* 和 *Max* 是指连续 30 个 ADU 的传输指纹数据量上下边界.本节不考虑 30 个 ADU 这个加强条件,因此 Reed 方法中对单个 ADU 长度的修正公式为:

$$ADU\_R = Payload\_S_c / p - q \tag{10}$$

公式(10)中,*p* 为 TLS 头部增加的载荷参数,*q* 为 HTTP 头部增加的载荷参数,文献[20]中 *p*=1.0018,*q*=520.

因为本文的数据集是 Facebook 数据,而文献[20]是针对 Netflix 平台的,本文首先使用 2.2 节中的 Facebook 数据集进行了回归拟合训练,一共 14551 个 ADU,其中 70%做训练集,30%做测试集,得到参数为 *p*=1.003676129,*q*=589.48.

$$ADU\_R = Payload\_S_c / 1.003676129 - 589.48 \tag{11}$$

利用公式(11)计算样本的 *ADU_R*,再使用明文指纹计算残差 *x*=*ADU_F*−*ADU_R*.训练集的结果为,残差的均值为 0,方差 1901.87,标准差 43.61;测试集结果为,残差的均值 0.588,方差 1799.17,标准差 42.42,可见训练误差和测试误差很接近,因此采用该模型是可行的.

图 12 为样本残差 PDF,可以看到 Reed 方法修正后残差主要分布在-100 字节到 100 字节之间.

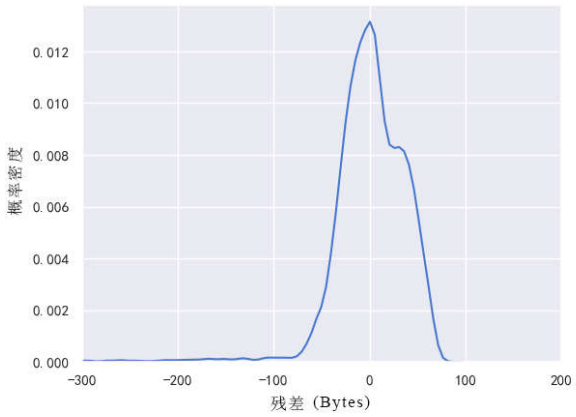



Fig.12 PDF of the ADU Length Residual with Reed method

图 12 Reed 方法修正后 ADU 长度残差的 PDF

由图 12 可见,Reed 方法获得的单个 ADU 长度残差分布可近似地看成正态分布, $\mu$ 为均值, $\sigma$ 为标准差,记作：$X\sim N(\mu, \sigma^2)$,可使用训练集残差的均值来无偏估计总体残差的均值,用训练集残差的标准差来无偏估计总体残差的标准差,则 $X\sim N(0, 43.61^2)$,残差在正负 3 倍标准差范围内的概率为 $P\{\mu-3\sigma<x<\mu+3\sigma\}=0.997$,即残差在[-130,130]区间内的概率为 99.7%.

利用公式(11)进行长度修正后再进行单个 ADU 匹配,已知 $Payload\_S_c$,计算得到 $ADU\_R$, 则这个 ADU 的明文长度 $ADU\_F$ 在[$ADU\_R$−130, $ADU\_R$+130]区间内的概率为 99.7%,定义该区间为 Reed 方法的匹配区间[c1,c2].使用 Reed 方法后进行单个 ADU 匹配方法为,通过上述方法算出匹配区间,匹配时指纹库中片段长度在匹配区间内的 ADU 为与之匹配的 ADU,其对应 ADU 明文指纹长度在匹配区间内的概率是 99.7%.

Reed 方法的匹配区间为[$c1,c2$] ,在匹配区间内任意明文指纹长度和修正值匹配的事件 A 的概率为 $P(A)=\int_{C1}^{C2} f(x)dx$ ,$f(x)$为图 11 所示 ADU 的概率密度函数,为简化计算,可以把匹配区间内的概率设为相等的一条直线, $x_0$ 为 ADU 长度分布的均值,简化公式为 $P_{Reed}(A)\approx f(x_0)\times(c2-c1)$ ,则使用 Reed 方法进行修正后匹配概率为：

$$P_{Reed}(A)\approx f(x_0)\times(c2-c1)=f(x_0)\times 261 \tag{12}$$

其中 $c2$-$c1$=261,是正态分布假设下匹配区间的范围.图 13 为该计算方法的示意图.

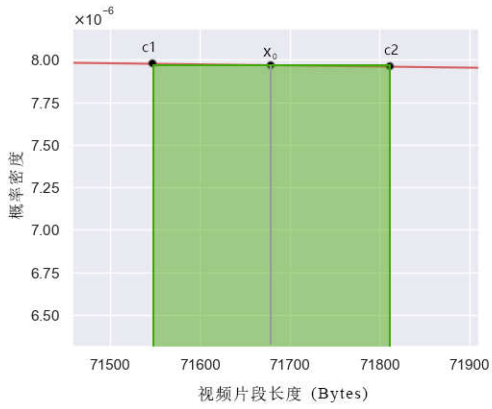



Fig.13 Schematic diagram of matching probability calculation

图 13 匹配概率简化计算示意图

根据 3.1 中模拟的测试视频指纹库的构建参数,可以得 $P_{Reed}(A) \approx f(x_0) \times 261 = 2.062 \times 10^{-3}$.

### 3.3 加密视频识别方法

加密视频的指纹是由每个视频 ADU 的长度及这些 ADU 传输的先后顺序构成的,识别是将待匹配的 ADU 长度修正值与指纹库中的明文长度按顺序使用 3.2 节的匹配算法进行匹配,如果有连续的 $k$ 段 ADU 匹配成功,则认为识别出了加密视频,我们称视频识别的过程为 ***k* 段匹配**.在一次匹配过程,假设明文指纹库中的一个视频明文指纹有 $j$ 个 ADU,观测到的加密视频传输指纹含有 $i$ 个 ADU,加密视频的 ADU 长度经过 HHTF 方法或者 Reed 方法复原后为 $x_1 \ldots x_i$ ,采用 $k(k \leqslant i$ 并且 $k \leqslant j)$个连续 ADU 匹配的方法来匹配,即如果 $i$ 个加密传输 ADU 中有 $k$ 个 ADU 和明文指纹的 $k$ 个 ADU 长度和顺序都匹配,则为完成了视频的 $k$ 段匹配.加密视频识别使用 $k$ 段匹配,关键参数 $k$ 需要根据评估指标在识别算法实施前确定.

定义事件 $E$ 为 ADU 个数为 $j$ 的明文指纹和 ADU 个数为 $i$ 的密文传输指纹 $k$ 段匹配成功,则事件 $E$ 的概率 $P(E)$为

$$P(E) = (i-k+1) \times (j-k+1) \times P(A)^k, \tag{13}$$

公式(13)中 $P(A)$为任意明文指纹长度和修正值 x 匹配事件 A 的概率.

$k$ 段匹配只是加密视频识别方法,匹配结果必然会存在误差,该方法要能在大型的指纹库场景中应用,必须对识别结果的各项指标进行全面评估,然后根据评估值确定 $k$ 的取值,只有指标达到要求的方法才能应用到实际中.

### 3.4 加密视频识别方法评估指标的理论计算

在加密视频识别算法评估中,使用准确率,查准率,查全率,假阳率可以全面评价算法的有效性,在实际应用中,$k$ 越大必然识别结果越准确,但是 $k$ 值大也意味需要采集连续传输且分辨率不变的 ADU 数量多,实际中采集到满足条件数据的可能性小,方法的可用性就差.所以对加密视频识别方法的评估需要求出满足准确率,查准率,查全率,假阳率这四个指标的最小 $k$ 值.

本节首先给出准确率,查准率,查全率,假阳率的理论评估值,并根据评估值确定 $k$ 的理想取值.然后在大型明文指纹库中测试,将理论值和测试对比验证方法的有效性,

假设明文指纹库中有 $t$ 个视频,一个待匹配加密视频和明文指纹库内 $t$ 个视频匹配过程中有 $s(s \geqslant 1)$个明文视频指纹 $k$ 段匹配成功,则事件 $E$ 的概率也可以表示为:

$$P(E)=\frac{s}{t} \tag{14}$$

将公式(13)代入公式(14),得到 $P(E)=\frac{s}{t}=(i-k+1)\times(j-k+1)\times P(A)^k$ (15)

其中$P(A)$为 ADU 长度均值$x_0$发生匹配事件的概率,假设待匹配视频的明文指纹一定在明文指纹库中,$s\geq 1$,因此 $P(E)\geq\frac{1}{t}$.

**准确率**：$A=\frac{TP+TN}{TP+TN+FP+FN}$,本文实验中待匹配视频的明文指纹在指纹库里,而且必然只对应一个明文指纹,所以 *TP*＝ 1 ； 其余不被匹配上的 *t-s* 个视频为 *TN*,代入准确率公式,得到：

$$A=\frac{TP+TN}{TP+TN+FP+FN}=\frac{1+(t-s)}{t}=1+\frac{1}{t}-P(E)=1+\frac{1}{t}-(i-k+1)\times(j-k+1)\times P(A)^k \tag{16}$$

**查准率**: $P=\frac{TP}{TP+FP}$,因为有 *s* 个明文视频指纹和待匹配视频 k 段匹配成功,*TP*+*FP*=*s*,代入查准率公式,得到：

$$P=\frac{TP}{TP+FP}=\frac{1}{s}=\frac{1/t}{s/t}=\frac{1}{t\times P(E)}=\frac{1}{t\times(i-k+1)\times(j-k+1)\times P(A)^k} \tag{17}$$

**查全率**： $R=\frac{TP}{TP+FN}$

查全率可以根据视频 ADU 的匹配概率推算出.

Reed 方法中一个待匹配视频和其相对应的明文指纹视频匹配时,待匹配 ADU 和其对应明文指纹匹配的概率是 99.7%,*k* 个连续 ADU 和它们对应的明文指纹都匹配的概率是 $0.997^k$,则 *k* 段 ADU 不能完全和它们对应的明文指纹匹配的概率是 $1-0.997^k$,*i* 个 ADU 中有*(i-k+1)*个连续的 *k* 段 ADU,这些 *k* 段 ADU 和它们相对应的明文指纹都不匹配的概率是 $(1-0.997^k)^{(i-k+1)}$,因此一个视频含有 *i* 个 ADU,和它对应的明文指纹视频可以 *k* 段匹配的概率为 $1-(1-0.997^k)^{(i-k+1)}$,即使用 Reed 方法的查全率为 $R_{Reed}=1-(1-0.997^k)^{(i-k+1)}$.当 *k* 较小, *i* 比 *k* 大很多的情况下,$R_{Reed}$接近 1,也就是如果待匹配视频的 ADU 数目较多,但是只使用较少的视频 ADU 去匹配,则查全率接近 1.

如果使用 HHTF 方法对加密数据进行复原,同理得到 $R_{HHTF}=1-(1-1^k)^{(i-k+1)}=1$.

由上述分析可见,使用 Reed 方法复原 ADU 长度指纹后查全率接近 1,使用 HHTF 方法复原 ADU 长度指纹后查全率等于 1.根据查全率的公式,得到 *TP*+*FN*=*TP*,即 *FN=0*.

**假阳率**： $FPR_{Reed}=\frac{FP}{FP+TN}$,因为 *s* 个被认定为匹配的明文视频中,只有 1 个是真正的匹配视频,其余 *s-1* 个视频为 *FP*,即 *FP*=*s-1*； 同样因为明文指纹库的所有 t 个视频中,只有 1 个是真正的匹配视频 *TP=1*,通过查全率已经推导出 *FN=0*,因为 *TP*+*FP*+*FN*+*TN*=*t*,所以 *FP*+*TN*=*t*−1,代入假阳率的公式,得到：

$$FPR_{Reed}=\frac{FP}{FP+TN}=\frac{s-1}{t-1},$$

大型指纹库中的视频数目远远大于 1,假阳率可以简化为：

$$FPR_{Reed}\approx\frac{s-1}{t}=\frac{s}{t}-\frac{1}{t}=P(E)-\frac{1}{t}=(i-k+1)\times(j-k+1)\times P(A)^k-\frac{1}{t}, \tag{18}$$

将 $P_{HHTF}(A)\approx 7.9\times10^{-6}$, $P_{Reed}(A)\approx f(x_0)\times 261=2.062\times10^{-3}$ 代入公式(16)—公式(18),并将测试指纹库中的 $t$=205677,$i$=280,$j$=426 代入,分别使用 2 个连续 ADU 匹配（$k$=2）,3 个连续 ADU 匹配（$k$=3）,可计算得到在这两种匹配长度下,分别使用 HHTF 方法和 Reed 方法修正 ADU 长度后,在大型指纹库进行视频匹配时的理论结果如表 1 所示.

**Table 1** Theoretical comparison of continuous ADUs matching results

**表 1** 连续 ADU 匹配结果理论比较

| 长度指纹修正方法 | $k$ | 准确率 | 查准率 | 查全率 | 假阳率 |
|---|---|---|---|---|---|
| HHTF | 2 | 99.9997% | 65.70% | 100% | $2.54\times10^{-6}$ |
| | 3 | **100%** | **100%** | **100%** | **0%** |
| Reed | 2 | 49.58% | $9.64\times10^{-6}$ | 100% | 50.41% |
| | 3 | 99.89% | 0.47% | 100% | 0.10% |

由表 1 可见,使用 HHTF 方法修正 ADU 长度后进行视频识别,只需要 3 个连续 ADU 就可以达到准确率、查准率、查全率为 100%,假阳率为 0.

### 3.5 加密视频识别方法在大型模拟指纹库中的实测结果和分析

为了验证表 1 理论评估值的正确性,用真实数据在大型模拟指纹库中进行了匹配识别,分别使用 2 个和 3 个连续 ADU 匹配,得到了 277 个真实视频在二十万级模拟指纹库中匹配的结果样例数如表 2：

**Table 2** Results from continuous ADU matching experiments in a large simulated fingerprint database

**表 2** 大型模拟指纹库中连续 ADU 匹配得到的结果

| 长度指纹修正方法 | $k$ | TP | FP | FN | TN |
|---|---|---|---|---|---|
| HHTF | 2 | 277 | 2404 | 0 | 56969848 |
| | 3 | 277 | 0 | 0 | 56972252 |
| Reed | 2 | 277 | 18157668 | 0 | 38814584 |
| | 3 | 277 | 1407324 | 0 | 55564928 |

将表 2 结果代入准确率、查准率、查全率和假阳率的公式(1)—公式(4),可以得到表 3 的实验结果.

**Table 3** Results of continuous ADU matching experiments in a large simulated fingerprint database

**表 3** 大型模拟指纹库中连续 ADU 匹配实验结果

| 长度指纹修正方法 | $k$ | 准确率 | 查准率 | 查全率 | 假阳率 |
|---|---|---|---|---|---|
| HHTF | 2 | 99.9958% | 10.33% | 100% | $4.22\times10^{-5}$ |
| | **3** | **100%** | **100%** | **100%** | **0%** |
| Reed | 2 | 68.1291% | $1.53\times10^{-5}$ | 100% | 31.87% |
| | 3 | 97.5298% | 0.0197% | 100% | 2.47% |

对比表 1 和表 3 的结果可见,理论分析结果和在大型模拟指纹库中的实测结果很接近,有些差别是因为,理论分析为了简化使用了 ADU 长度均值的匹配概率,而实测中使用的是 ADU 长度的真实值去匹配.

对实验结果进行进一步比较分析,可确定 HHTF 修正方法应用到大规模指纹库中进行加密视频识别算法的有效性.

（1）准确率：使用 2 个连续 ADU 进行匹配获得的准确率 Reed 方法较低,HHTF 方法较高,使用 3 个连续 ADU 进行匹配后准确率都较高,其中使用 HHTF 方法准确率非常接近 100%,这说明准确率指标在大型数据库中达标并不困难,该指标对不同算法的区分度不够.

（2）查准率：查准率指标差别很大,总体上使用 HHTF 方法的查准率高于使用 Reed 方法,使用 3 个连续 ADU 匹配后,HHTF 方法查准率为 100%,但是使用 Reed 方法查准率很低,这是因为在大型指纹库中,使用 Reed 方法后得到的 FP 样例远远大于 HHTF 方法,这导致使用 Reed 方法的视频匹配在大型数据库中查准率差,由此可见大型指纹库中的查准率是一个重要的有区分度的指标.

（3）查全率：两种方法的查全率都很高,这说明查全率指标对设计合理的识别算法来说并没有区分度,现有文献大都以查全率作为评估指标并不合理.

（4）假阳率：Reed 方法的假阳率远大于 HHTF 方法,当使用 3 个连续 ADU 识别时,HHTF 方法的假阳率指标为 0,而 Reed 方法的假阳率仍然不能满足识别要求.这也是因为使用 Reed 方法后得到 FP 样例在大型数据库中数值非常大,导致假阳率高,由此可见大型指纹库中的假阳率是一个重要的有区分度的指标.

（5）$k$=3 时的所有指标都比 $k$=2 时好,即增加 $k$ 值可以提高查准率,降低假阳率,但是 $k$ 越大需要的 ADU 个数越多,在实际应用中越难采集到需要的数据量,因此 $k$ 的取值不宜过多,由表 1 和表 3 的结果可见,使用 HHTF 方法后,只需要 3 段匹配就可以满足在二十万级指纹库中的识别需求,而文献[20,21]中都提到 ADU 个数要求为 30,文献[23,24]提到 3 分钟的数据,相当于 Facebook 的 90 个 ADU.对比之下,HHTF 方法应用后需要的 ADU 个数大大减少,提高了方法的可用性.

由上述对准确率、查准率、查全率和假阳率的分析比较可见,这两种方法对 ADU 进行修正后进行视频匹配,准确率和查全率指标比较接近,但是查准率和假阳率在大型数据库中指标差别很大.

HHTF 方法查准率和假阳率指标优于 Reed 方法是因为 Reed 方法对 ADU 长度的复原不够精确,为了保证视频能够被识别出来,Reed 方法对单个 ADU 需要较大的匹配区间,但是匹配区间增大也会导致 $FP$ 数目增加.在大型指纹库场景中,ADU 数据多,同样大的匹配区间内存在更多的长度近似的 ADU,因此 Reed 方法的 $FP$ 数目在大型指纹库中急剧增加.查准率的公式为 $P=\frac{TP}{TP+FP}$,在本次测试中,$TP$=277,因此 $FP$ 越大,查准率越小.假阳率的公式为 $FPR=\frac{FP}{FP+TN}$,由于 $FP$+$TN$+$TP$+$FN$=总的匹配次数,其中 $TP$=277,$FN$=0, 随着指纹库规模的增大,总的匹配次数必然增大,在全匹配情况下是与指纹库规模相关的定量,所以 $FP$+$TN$ 也是定量,随着 FP 的增加,假阳率也会增加.由此可见,ADU 长度精准复原方法 HHTF 是我们可以在大型指纹库中准确识别视频的基础.

综合看来,HHTF 方法指标远远优于 Reed 方法.在本文使用的二十万级别大型指纹库中,使用 HHTF 方法复原 ADU 长度后,只需要 3 个连续 ADU（Facebook 视频为 6 秒播放数据）就可以准确识别出加密视频,准确率、查准率、查全率为 100%,假阳率为 0,完全达到实际应用需求的指标要求.

### 3.6 加密视频识别方法在小型指纹库中的实测结果和分析

为了进一步比较两种修正方法应用于不同规模指纹库的效果,本节给出在小型真实指纹库中分别使用 HHTF 方法和 Reed 方法进行修正后得到的实验结果.使用真实的 277 个视频构成一个小型指纹库,分别用 2 个和 3 个连续 ADU 匹配,得到了 277 个真实视频在真实指纹库中匹配的结果如表 4 所示：

**Table 4** Results from continuous ADU matching experiments in a small real fingerprint database

**表 4** 小型真实指纹库中连续 ADU 匹配得到的结果

| 长度指纹修正方法 | k | TP | FP | FN | TN |
|---|---|---|---|---|---|
| HHTF | 2 | 277 | 2 | 0 | 76450 |
| | 3 | 277 | 0 | 0 | 76452 |
| Reed | 2 | 277 | 23152 | 0 | 53300 |
| | 3 | 277 | 1964 | 0 | 74488 |

将表 4 结果代入准确率、查准率、查全率和假阳率的公式(1)—公式(4),可以得到表 5 的实验结果：

**Table 5** Results of continuous ADU matching experiments in a small real fingerprint database

**表 5** 小型真实指纹库中连续 ADU 匹配实验结果

| 长度指纹修正方法 | k | 准确率 | 查准率 | 查全率 | 假阳率 |
|---|---|---|---|---|---|
| HHTF | 2 | 99.9974% | 99.28% | 100% | $2.62\times10^{-5}$ |
| | **3** | **100%** | **100%** | **100%** | **0%** |
| Reed | 2 | 69.83% | 1.18% | 100% | 30.28% |
| | 3 | 97.44% | 12.36% | 100% | 2.57% |

由表 5 的结果可以看到:

(1)HHTF 方法和 Reed 方法修正后查全率指标都很理想,但事实上只有 HHTF 的四项指标全部符合要求.这证明了查全率对算法的区分度不高,现有成果中最广泛使用的查全率指标不能全面评估算法,只有四个指标同时达到理想值才能判断算法是可用的.

(2) 在表 5 的结果中,使用 HHTF 方法,只要 2 个连续的 ADU 就可以达到理想的识别指标,但是对比表 3 的实验结果可以看到,当指纹库规模达到二十万数量级时,2 个连续的 ADU 进行匹配查准率只有 10.33%,会有大量其它视频被误识为识别视频,必须使用 3 个连续 ADU 进行匹配.这说明了,随着指纹库规模的增加,FP 事件必然会上升,因此对小型指纹库适用的识别参数在大型指纹库里未必适用,只有直接在大型指纹库中进行算法验证,结果才具有可信度.

### 3.7 实验结果通用性验证

上述实验证明了必须使用大型指纹库才能真正验证算法的可行性.由于无法得到真实的大型明文指纹库,本文 3.1 节基于统计学原理,使用 277 个 Facebook 视频的 77802 个 ADU 长度统计特征,构建了一个模拟的大型视频指纹库进行验证.

为了验证实验结果与模拟指纹库所使用的真实视频无关,本节将 277 个视频分成不相交的 2 组视频集,第 1 组包括 139 个 Facebook 视频,含有 40215 个 ADU,第 2 组包括 138 个 Facebook 视频,含有 37587 个 ADU,按照同样的方法,先分别统计 ADU 长度 PDF,再基于 AUD 长度的 PDF,按照 3.1 所述的方法构造两个大型模拟数据库,除了完全不相交的两组真实视频,所有模拟视频构造过程中,ADU 长度遵循真实的 Facebook 视频 ADU 长度 PDF,各视频长度使用了一定的随机变化,因此这两个大型指纹库是不同的.使用同样的匹配方法,得到两组实验结果.将这两组实验结果与 3.5 中的实验结果全部列入表 6,对三个不同大型模拟指纹库匹配实验结果比较.

可以看到,用来构造模拟指纹库的样本不同,样本 ADU 个数不同,模拟出的指纹库规模接近,参数 $k$ 相同的情况下,各项指标差别非常小,这些微小的差别完全可以视为样本个体差异引起的,对总体的统计结论是一致的.

**Table 6** Comparison of the results of three large-scale simulated database matching experiments

**表 6** 三个不同大型模拟指纹库匹配实验结果比较

| 修正方法 | 样本视频数 | 样本 ADU 数 | 模拟指纹库视频数 | $k$ | 准确率 | 查准率 | 查全率 | 假阳率 |
|---|---|---|---|---|---|---|---|---|
| HHTF | 139 | 40215 | 177939 | 2 | 99.9959% | 12.00% | 100% | $4.12\times10^{-5}$ |
| HHTF | 138 | 37587 | 177738 | 2 | 99.9958% | 11.85% | 100% | $4.19\times10^{-5}$ |
| HHTF | 277 | 77802 | 205677 | 2 | 99.9958% | 10.33% | 100% | $4.22\times10^{-5}$ |
| HHTF | 139 | 40215 | 177939 | 3 | 100% | 100% | 100% | 0% |
| HHTF | 138 | 37587 | 177738 | 3 | 100% | 100% | 100% | 0% |
| HHTF | 277 | 77802 | 205677 | 3 | 100% | 100% | 100% | 0% |
| Reed | 139 | 40215 | 177939 | 2 | 67.0524% | $1.71\times10^{-5}$ | 100% | 32.95% |
| Reed | 138 | 37587 | 177738 | 2 | 68.6022% | $1.79\times10^{-5}$ | 100% | 31.40% |
| Reed | 277 | 77802 | 205677 | 2 | 68.1291% | $1.53\times10^{-5}$ | 100% | 31.87% |
| Reed | 139 | 40215 | 177939 | 3 | 97.5414% | 0.0229% | 100% | 2.46% |
| Reed | 138 | 37587 | 177738 | 3 | 97.6198% | 0.0236% | 100% | 2.38% |
| Reed | 277 | 77802 | 205677 | 3 | 97.5298% | 0.0197% | 100% | 2.47% |

由表 6 结果可以看到,只要样本量足够大,样本选择具有独立性和代表性,使用不同的真实样本构造模拟指纹库,不影响本文算法的实验结果的通用性.

## 4 结束语

本文提出了一个大型指纹库场景中加密视频识别的方法. 首次将 HTTP 头部特征和 TLS 片段特征作为 ADU 长度复原的拟合特征, 提出了一个 ADU 长度精准复原方法 HHTF, 对于满足要求的密文数据, 可从单个视频 ADU 的传输长度准确复原出明文ADU长度, 然后通过理论分析和模拟的大规模指纹库实验证明了, 将HHTF方法应用

于 Facebook 的加密视频识别,在二十万级指纹库中识别视频达到准确率、查准率、查全率为 100%,假阳率为 0 只需要 3 个连续的 ADU,所需 ADU 个数是已有研究的十分之一,这大大降低了对密文数据采集需求.

本文对视频识别方法的评估使用准确率、查准率、查全率和假阳率这四个指标,可以全面反映方法的适用性,目前已有的加密视频识别方法评估都使用了区分度不高的查全率,但是都回避了在大型指纹库中的查准率和假阳率指标,导致已有的研究成果无法应用于大型指纹库中.本文的成果填补了这一空白,具有很强的应用价值.

本文的关键技术在于基于 TLS1.2 加密及传输过程原理提出了 ADU 长度精准复原算法 HHTF,在对数据预处理的时候充分考虑了网络传输的中的各种复杂现象,保证了待匹配数据的准确性,从而能提取出关键特征;而现有成果的研究重点都是在后期的匹配算法上,没有考虑网络传输环境的复杂性,无法提取出数据关键特征,因此无法精准复原视频指纹,导致在大型数据库场景中的性能无法保证.

本文利用了 ADU 加密传输过程中的协议规范将加密传输的 ADU 长度精准复原,但是因特网上协议规范会不断更新,现在已有一些网站使用 TLS1.3 协议进行加密传输,要想保持算法结果的精确性,就需要提取新的特征值.此外,使用基于 UDP 的 QUIC 协议进行加密传输也是发展趋势之一,对 QUIC 协议的特征提取是识别 QUIC 协议加密传输视频的关键,这些都是未来本领域的研究点.

**References**:

[1] ZHAO Bo, GUO Hong, LIU Qin- Rang, & WU Jiang- Xing. Protocol Independent Identification of Encrypted Traffic Based on Weighted Cumulative Sum Test.Journal of Software, 2013 , 24(6):1334-1345.

[2] Velan, P., Čermák, M., Čeleda, P., & Drašar, M. A survey of methods for encrypted traffic classification and analysis. International Journal of Network Management, 2015, 25(5): 355-374. [doi: 10.1002/nem.1901]

[3] Xiang C, Chen Q, Xue M, & Zhu H. APPCLASSIFIER: Automated App Inference on Encrypted Traffic via Meta Data Analysis. In: Proc. of the 2018 IEEE Global Communications Conference (GLOBECOM). Piscataway, NJ: IEEE, 2018.1-7. [doi: 10.1109/GLOCOM.2018.8647508]

[4] Taylor, V. F., Spolaor, R., Conti, M., & Martinovic, I.. Robust smartphone app identification via encrypted network traffic analysis. IEEE Trans. on Information Forensics and Security, 2017, 13(1): 63-78. [doi: 10.1109/TIFS.2017.2737970]

[5] Anderson B, Paul S, & McGrew D. Deciphering malware's use of TLS (without decryption). Journal of Computer Virology and Hacking Techniques, 2018,14: 195-211. [doi: 10.1007/s11416-017-0306-6]

[6] Anderson B, McGrew D. Identifying encrypted malware traffic with contextual flow data. Proc. of the 2016 ACM workshop on artificial intelligence and security. New York: ACM, 2016: 35-46. [doi: 10.1145/2996758.2996768]

[7] Ahmed M E, Ullah S, & Kim H. Statistical Application Fingerprinting for DDoS Attack Mitigation. IEEE Trans. on Information Forensics and Security, 2019, 14(6): 1471-1484. [doi: 10.1109/TIFS.2018.2879616]

[8] Biernacki A. Identification of adaptive video streams based on traffic correlation. Multimedia Tools and Applications, 2019,78(13):18271-18291. [doi: 10.1007/s11042-019-7183-6]

[9] Li F, Chung J W, & Claypool M. Silhouette: Identifying youtube video flows from encrypted traffic. Proc. of the 28th ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio and Video. New York: ACM, 2018: 19-24.

[10] Tang Shuang, Qin XW, & Wei G. Network-Based Video Quality Assessment for Encrypted HTTP Adaptive Streaming. IEEE Access, 2018, 6:56246-56257. [doi: 10.1109/ACCESS.2018.2872932]

[11] Orsolic I, Skorin-Kapov L, & Suznjevic M. Towards a Framework for Classifying YouTube QoE Based on Monitoring of Encrypted Traffic. International Young Researcher Summit on Quality of Experience in Emerging Multimedia Services (QEEMS 2017). 2017[2019-6-11] https://www.fer.unizg.hr/_news/71871/qeems_paper.pdf

[12] Orsolic, I., Pevec, D., Suznjevic, M., & Skorin-Kapov, L. A machine learning approach to classifying YouTube QoE based on encrypted network traffic. Multimedia tools and applications, 2017, 76(21): 22267-22301. [doi: 10.1007/s11042-017-4728-4]

[13] Dimopoulos, G., Leontiadis, I., Barlet-Ros, P., & Papagiannaki, K. Measuring video QoE from encrypted traffic. In: Proc. of the 2016 Internet Measurement Conference. New York: ACM, 2016: 513-526. [doi: 10.1145/2987443.2987459]

[14] Pan W, Cheng G, Wu H, & Tang Y. Towards QoE assessment of encrypted YouTube adaptive video streaming in mobile networks. In: Proc. Of the2016 IEEE/ACM 24th International Symposium on Quality of Service(IWQoS). Piscataway, NJ: IEEE, 2016.1-6. [doi: 10.1109/IWQoS.2016.7590437]

[15] CISCO. Cisco Annual Internet Report (2018-2023) White Paper．[2019-02-17]．https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html

[16] Gu XD, Yang M, Luo JZ, & Jiang P. Website Fingerprinting Attack Based on Hyperlink Relations. Chinese Journal of Computers, 2015, 38(04):833-845(in Chinese). [doi: 10.3724/SP.J.1016.2015.00833]

[17] Panchenko A, Niessen L, Zinnen A, & Engel T. Website fingerprinting in onion routing based anonymization networks. Proc. of the 10th annual ACM workshop on Privacy in the electronic society. New York: ACM, 2011: 103-114. [doi: 10.1145/2046556.2046570]

[18] Cai X, Zhang X C, Joshi B, & Johnson R.. Touching from a distance: Website fingerprinting attacks and defenses. In: Proc. of the 2012 ACM conference on Computer and communications security. New York: ACM, 2012: 605-616. [doi: 10.1145/2382196.2382260]

[19] Terrell J, Jeffay K, Smith F D, Gogan J, & Keller J. Passive, Streaming Inference of TCP Connection Structure for Network Server Management. In: Proc. Of the International Workshop on Traffic Monitoring and Analysis. Springer, Berlin, Heidelberg, 2009: 42-53. [doi: 10.1007/978-3-642-01645-5_6]

[20] Reed A, Kranch M. Identifying https-protected netflix videos in real-time. In: Proc. of the Seventh ACM on Conference on Data and Application Security and Privacy. New York: ACM, 2017: 361-368. [doi: 10.1145/3029806.3029821]

[21] Reed A, Klimkowski B. Leaky streams: Identifying variable bitrate DASH videos streamed over encrypted 802.11n connections. In: Proc of the 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC). Piscataway, NJ: IEEE,2016: 1107-1112.

[22] Stikkelorum M. I Know What You Watched: Fingerprint Attack on YouTube Video Streams. In: 27th Twente Student Conference on IT. Enschede, Netherlands. 2017． https://pdfs.semanticscholar.org/2015/26efeb7206e2704b8db46985e4fcb0b93e55.pdf

[23] Gu J, Wang J, Yu Z, & Shen K. Walls have ears: Traffic-based side-channel attack in video streaming. Proc. Of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications. Piscataway, NJ: IEEE, 2018: 1538-1546. [doi: 10.1109/INFOCOM.2018.8486211]

[24] Gu J, Wang J, Yu Z, & Shen K. Traffic-Based Side-Channel Attack in Video Streaming. IEEE Trans. on Networking, 2019, 27(3):972-985. [doi: 10.1109/TNET.2019.2906568]

[25] Mitra G, Vairam P K, SLPSK P, Chandrachoodan N, & Kamakoti V. White Mirror: Leaking Sensitive Information from Interactive Netflix Movies using Encrypted Traffic Analysis. Proc. of the 2019 ACM SIGCOMM Conference Posters and Demos, Part of SIGCOMM 2019, 122-124, August 19, 2019. [doi: 10.1145/3342280.3342330]

[26] Schuster R, Shmatikov V, & Tromer E. Beauty and the burst: Remote identification of encrypted video streams. IN: Proc. Of the 26th {USENIX} Security Symposium ({USENIX} Security 17). PeerJ,San Diego,2017: 1357-1374.

[27] Sodagar I. The MPEG-DASH standard for multimedia streaming over the Internet. IEEE Multimedia, 2011, 18(4):62-67.[doi: 10.1109/MMUL.2011.71]

[28] Pantos R, May W. RFC8216: HTTP Live Streaming. Fremont, CA: IETF, 2017. https://tools.ietf.org/html/rfc8216.

[29] Dierks T, Rescorla E, RFC5246: The Transport Layer Security (TLS) Protocol Version 1.2. Fremont, CA: IETF, 2008. https://tools.ietf.org/html/rfc5246

## 附中文参考文献:

[1] 赵博,郭虹,刘勤让,邬江兴.基于加权累积和检验的加密流量盲识别算法.软件学报,2013,24(6):1334-1345.

[16] 顾晓丹,杨明,罗军舟,蒋平.针对 SSH 匿名流量的网站指纹攻击方法.计算机学报,2015,38(4):833-845.