面向内容监管的 P2P-TV 音视频数据还原与在线检测方法研究 *

康 浩, 何 速, 姜志宏, 张 鑫, 樊鹏翼
(国防科学技术大学 信息系统与管理学院, 长沙 410073)

摘 要: 为实现对 P2P-TV 应用的实时内容检测,简要介绍了 P2P-TV 监控系统对 P2P-TV 平台与频道的精细识别,针对 PPTV 采用 ASF 流媒体格式进行数据流传输、节点之间通过 UDP 协议获取数据,在精确识别出平台与频道的基础上,识别出数据传输过程中的 A/V 数据包,获知 A/V 数据包的序号、A/V 数据的长度及起始终止位置,通过在线将 A/V 数据提取并还原为媒体文件并进行内容检测。

关键词: P2P-TV; 数据还原; 内容检测
中图分类号: TP393 **文献标志码**: A **文章编号**: 1001-3695(2012)01-0187-04
doi:10.3969/j.issn.1001-3695.2012.01.053

Research on audio/video data recovery and online detection method of P2P-TV for content monitoring

KANG Hao, HE Su, JIANG Zhi-hong, ZHANG Xin, FAN Peng-yi
(*College of Information System & Management, National University of Defense Technology, Changsha* 410073, *China*)

Abstract: For the purpose of online content monitoring on P2P-TV, this paper firstly made a short introduction to the monitoring system of P2P-TV. Then according to the fact that PPTV utilized ASF to transmit data and that peers obtained data from each other by UDP method, based on the accurate identification of platform and channel of P2P-TV, this paper identified the audio/video data packets in the process of data transmission, and then acquired the serial numbers of those packets, the length of audio/video data, the beginning and ending of audio/video data in each packet, finally extracted the audio/video data online, recovered them to media files and implemented content detection.

Key words: P2P-TV; data recovery; content detection

0 引言

对等网络 (peer-to-peer, P2P) 是近几年新兴的网络技术。相对于传统的客户机/服务器(client/server, C/S)模式,P2P 模式一个非常显著的特点就是节点无须完全依赖集中式服务器资源,各节点之间可以直接进行通信。每个节点具有相同的地位,既可以请求服务,也可以提供服务,在从其他节点处获取数据流的同时也向其他节点发送数据流,扮演着 C/S 模式中客户机和服务器的双重角色。流媒体(media streaming)技术允许用户通过网络一边下载一边播放多媒体数据,不需要等待文件全部下载完毕后再使用。而 P2P 流媒体技术是借鉴 P2P 文件共享的基本原理,结合流媒体技术的优点,在保证系统实时性的前提下,实现了视频与音频等媒体信息实时的分发与传播。

目前国内比较流行的 P2P-TV 应用有 PPTV(PPLive)[1]、PPStream[2]、UUSee[3]、Sopcast[4]、QQLive[5] 等。由于大部分 P2P-TV 应用都采用私有协议,容易脱离监管的范围;又加上 P2P 网络本身的脆弱性,这些应用很容易成为一些不法分子的攻击目标,成为传播非法信息的渠道。目前 P2P-TV 应用面临的主要威胁在于:a)色情、暴力等非法内容在 P2P-TV 中传播;b)盗版或未经授权的多媒体信息通过 P2P-TV 传播。

因此对 P2P-TV 的监管迫在眉睫,刻不容缓。国内外许多研究机构针对 P2P-TV 的监管技术展开了大量研究,主要集中在 P2P-TV 的网络测量和流量识别[6~9] 两个方面。网络测量主要是挖掘 P2P-TV 的网络拓扑特性、流量特征以及用户行为特征,建立相关的行为模型;而流量识别则对网络流量进行识别与分类,实现 P2P-TV 的平台与频道等流量的精确检测。然而,一旦 P2P-TV 遭受插播或污染攻击,音/视频(audio/video, A/V)数据被窜改,上述两种方法都无法及时作出识别与警报,因此无法实现对 P2P-TV 播放内容实时、精确的检测。同时,由于各平台协议的私有性以及节点加入网络的随意性,内容检测成为当前 P2P-TV 监管技术的一大难点。

为实现对园区网内 P2P-TV 的监管,本课题组提出并设计了一套 P2P-TV 监控系统,如图 1 所示。本系统主要由数据采集、识别、异常检测和管控四个模块组成。数据采集模块主要通过旁路光纤和网卡在园区网入口上产生镜像流量,并从中采集本系统所需的各种网络流量;识别模块能够实时地从网络流量中检测出 P2P 视频组播流,对典型平台及其频道进行识别,并检测出网络流量中的未知 P2P-TV 流量;异常检测模块获取已识别的平台频道的 A/V 数据,将其还原为媒体文件,并对其

收稿日期: 2011-05-08; **修回日期**: 2011-07-07 **基金项目**: 国家"863"计划资助项目(2008AA01Z407)

作者简介: 康浩(1985-),男,陕西西安人,硕士研究生,主要研究方向为 P2P 网络(xianchangsha@126.com);何速(1986-),男,江西九江人,博士研究生,主要研究方向为 P2P 网络、文本挖掘.

进行异常内容检测,如发现异常及时进行预警;管控模块对于探测到的可疑流量,一方面产生报警信号提醒业务人员关注,另一方面借助人的参与,并依据预先定义的管控策略进行异常流量的过滤。

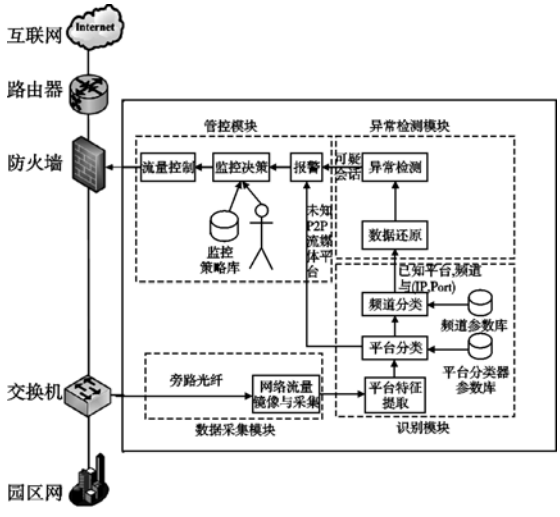

图 1 P2P流媒体精细识别与管控系统

异常检测模块主要采取两种方式进行异常内容检测:a)将视频数据还原,判断电视台台标是否被窜改;b)将音频数据还原,借助已有的音频转换软件将其转换为文本,进行异常内容检测。目前已有的 P2P 录像软件[9] 仅能够实现对用户本地播放的 P2P-TV 进行实时数据封装,不能满足本系统从镜像数据流中实现数据还原的需要。

1 P2P-TV 流量精细识别方法

园区网内不同用户会同时开启不同的网络应用,产生大量多样的流量。如何从复杂的网络流量中识别出 P2P-TV 流量是实现 P2P-TV 网络监管的基本前提,而同时识别出 P2P-TV 流量对应的平台与频道则有利于更加全面、准确的监管。本系统实现了对复杂网络流量中 P2P-TV 平台和频道的识别与分类。

1.1 P2P-TV 平台识别

各种基于 UDP 协议的 P2P-TV 应用通常会使用大小不同的数据包来交换 A/V 数据,因此可能会产生不同的包大小分布[10]。由于园区网内最大传输单元为 1 500 Byte,设置 30 个区间,每 50 Byte 一个区间。选择五个平台:PPTV、PPStream、QQLive、UUSee、Sopcast,每个平台随机选择一个频道,采集下行数据,每段数据的持续时长为 2 250 s。设时间窗口为 5 s,统计该段时间内大小落于每个区间内的所有包的总字节数占接收到的全部数据包的总字节数的比例。可得到 450 次统计结果,取其平均值,结果如图 2 所示。

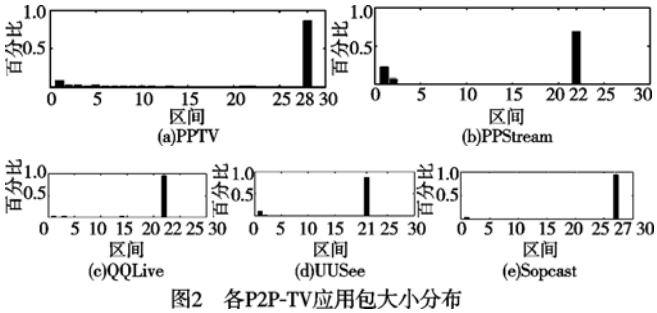

图2 各P2P-TV应用包大小分布

从图 2 可见,五个不同的 P2P-TV 应用有着不同的包大小区间分布。其中 PPStream 和 QQLive 两种应用在第 22 区间(1 050 ~ 1 100 Byte)都有很高的字节百分比,但实际上在这个区间它们有不同的包大小:PPStream 是1 069 Byte,而 QQLive 是 1 068 Byte。根据对五种 P2P-TV 流量的分析可以发现,每种应用都有几个主要的包大小分布区间,如表 1 所示。

表 1 五种 P2P-TV 应用的主要包大小分布区间

应用编号	名称	主要分布区间
1	PPTV	21 ~ 110, 199 ~ 205, 1378 ~ 1384
2	PPStream	21 ~ 110, 1069
3	QQLive	21 ~ 110, 113, 1068
4	UUSee	0 ~ 20,21 ~ 110, 1005 ~ 1025
5	Sopcast	21 ~ 110, 1320

以此为依据,本系统构建了一个识别特征向量 C-PSD,使用支持向量机(SVM)构造多分类器,可以快速地从复杂的网络流量中精确识别出不同的 P2P-TV 流量[11],即实现了平台识别。

1.2 P2P-TV 频道识别

从网络流量中识别出 P2P-TV 流量后,本系统利用频道签名实现对频道的识别。所谓频道签名,是指 P2P-TV 报文中可用于识别具体的 P2P-TV 平台与频道流量的签名特征。本系统通过对频道签名挖掘,得到了五种 P2P-TV 常见频道的签名特征[12]。表 2、3 给出了两种 P2P-TV 应用的五个频道的签名特征字符串。

表 2 PPTV 五个频道的签名特征字符串

频道名称	频道签名															
东方卫视	8c	07	5b	6c	54	55	5d	4e	89	43	53	17	da	a4	4b	c5
湖南卫视	6f	81	a0	a1	0f	ed	97	4c	af	29	29	e5	f4	e4	e6	57
江苏卫视	9f	b9	7a	61	83	e4	e0	49	8e	6c	18	81	96	ef	09	02
浙江卫视	7a	13	26	e4	11	23	58	4b	a0	60	ed	11	15	73	50	d6
深圳卫视	dd	cf	1f	c4	88	b9	d4	43	8b	19	96	25	c4	7e	2d	95

表 3 PPStream 五个频道的签名特征字符串

频道名称	频道签名									
东方卫视	15	00	43	00	00	9b	b0	75	46	00
湖南卫视	15	00	43	00	00	9b	17	23	fd	00
江苏卫视	15	00	43	00	00	89	bb	47	d7	00
浙江卫视	15	00	43	00	00	9b	f8	08	a9	00
深圳卫视	15	00	43	00	00	9b	f3	93	85	00

通过对 P2P-TV 平台与频道的识别,可实现园区网内精细到单个用户(IP,Port)、具体到某一平台某一频道的流量识别,以此作为实时还原 A/V 数据与进行内容检测的基础。

1.3 P2P-TV 流量中 A/V 数据的识别与定位

以 PPTV 为例,观看 PPTV 的用户节点通过 UDP 协议从其他节点处获取 A/V 数据,将获得的一块块不连续的数据在缓冲区中按照 A/V 数据播放的时间顺序排列好,再推向播放器部分进行播放。在实施 A/V 数据还原之前,需要经过以下步骤:a)识别出 A/V 数据包;b)获知 A/V 数据包的序号;c)找出 A/V 数据包中 A/V 数据的位置。

1.3.1 A/V 数据包的识别

当园区网内用户启动 PPTV 应用后,该节点首先与 Web 服务器建立 TCP 连接,获取版本更新、频道列表等信息。选择频

道后,该节点开放一个本地 UDP 端口,与 Tracker 服务器通信,以查询节目源和节点数量。然后向其他节点发送探测消息,发现活动节点后,建立连接依次交换节点列表、请求和传送数据块[13]。在此过程中,节点 UDP 端口收到的不仅有 A/V 数据包,也有大量的控制报文。通过统计分析,发现八种长度的数据包中包含 A/V 数据,如表 4 所示。

表 4　A/V 数据包的长度

编号	数据包长度/Byte			
1	241	243	245	247
2	1420	1422	1424	1426

1.3.2　A/V 数据包的序号分析

节点从其他节点处请求到 A/V 数据块后,按一定的顺序组装起来,当组装的 A/V 数据块达到指定数量后就会启动播放器。播放器不断地从本地 HTTP 流媒体服务器下载 A/V 数据,并采用播放器本身自带的缓存机制将 A/V 数据缓冲起来,当缓冲到一定数量后,所选节目画面就呈现在用户面前[13]。因此,在每个 A/V 数据包中,应该存在一个序号,能够使所有 A/V 数据包按播放的时间先后顺序排列起来。经统计验证发现,A/V 数据包的序号采用小尾端法,用 6 个字节表示。序号位置具体如表 5 所示。

表 5　A/V 数据包序号位置统计

数据包长度/Byte	241	243	245	247
序号位置	63 ~ 68	65 ~ 70	67 ~ 72	69 ~ 74
数据包长度/Byte	1420	1422	1424	1426
序号位置	63 ~ 68	65 ~ 70	67 ~ 72	69 ~ 74

1.3.3　A/V 数据包中 A/V 数据的起始位置与长度分析

A/V 数据包由两部分组成:包头和数据部分。包头长度为 42 Byte,数据部分位于第 43 Byte 之后。数据部分除了包含 A/V 数据外,也有诸如包序号之类的控制信息以及冗余数据,如图 3 所示。那么,需确定 A/V 数据的位置与长度。

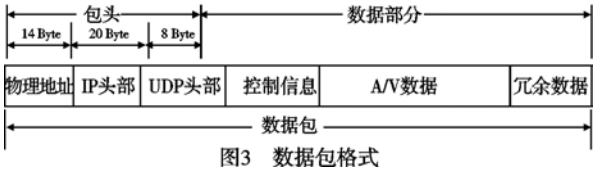

图3　数据包格式

经统计验证发现,A/V 数据的起始位置与 A/V 数据包长度是一一对应的关系,具体如表 6 所示。

表 6　视频数据起始位置与长度

数据包长度/Byte	241		243		245		247	
起始位置	77		79		81		83	
数据长度	144		144		144		144	
数据包长度/Byte	1420		1422		1424		1426	
起始位置	77	93	79	95	81	97	83	99
数据长度	1344	1328	1344	1328	1344	1328	1344	1328

长度为 241、243、245、247 Byte 的 A/V 数据包的后 21 个字节为冗余数据;长度为 1 420、1 422、1 424、1 426 Byte 的 A/V 数据包中,如果出现 82 00 00 11 5d(十六进制)的 ASF 格式中包头(packet header)的标志符[14],且 82 出现在第 93、95、97、99 个字节的位置,即标志着之前的 16 Byte 为冗余数据。

2　P2P-TV 数据流中 A/V 数据的实时还原

实时还原 P2P-TV 的数据流中的 A/V 数据,涉及以下四个方面的内容:网络数据包的识别与获取、数据包的重排序、A/V 数据的提取、A/V 文件的封装。以 PPTV 为例进行数据还原。鉴于音频数据还原与视频数据还原的思路和方法大致相同,这里只对视频数据还原作具体阐述。整个过程采用 C ++ 语言实现。

2.1　识别与获取视频数据包

从识别模块中得到校园网内正在收看 PPTV 电视直播的某一用户的 IP 地址及 UDP 端口,在 Wireshark 软件中设置 IP 地址、UDP 端口过滤,从镜像数据流中抓取满足表 3 所示的八种长度的数据包即为视频数据包。

2.2　重排序视频数据包

依据表 5,由视频数据包的长度可得知序号的位置,并计算其大小。利用序号对视频数据包进行重排序,形成一个有序的链表。

2.3　提取视频数据对象

ASF 格式文件的包(packet)存在于视频数据包中的视频数据中。包是容纳多媒体数据的容器,由头(header)和若干个负载(payload)组成,负载又由头和数据(data)组成[14],如图 4 所示。

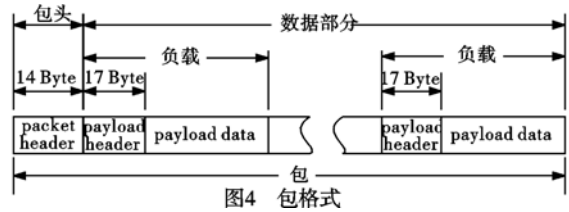

图4　包格式

同一包内的负载隶属于不同的数据对象(data object)。数据对象是包含视频数据的单元,由若干个负载组成。视频数据对象中的负载数据即为完全的视频信息。从排好序的视频数据包中,确定视频数据对象的起始位置并得到其包含的所有负载数据,最终形成一个视频数据对象链表。

2.3.1　确定视频数据对象的起始位置

可以利用视频数据对象的第一个负载头来判定其位置。在 PPTV 数据传输过程中,负载头是一段 17 Byte 的字符串(十六进制)。举例如图 5 所示。

02	94	00 00 00 00	08	7a 0b 00 00	74 5a 88 01	7f 08

图5　负载头示例

各个字节代表的含义如表 7 所示。

表 7　负载头各字节的含义

位置	数据	含义	备注
1	02	数据对象类型	02 表示非关键视频对象,82 表示关键视频对象
2	94	数据对象序号	从 00 到 ff,每 256 个数据对象一个循环
3 ~ 6	00 00 00 00	当前负载起始位置相对于所属对象起始位置的偏移	为 0 时,表示是视频数据对象的第一个负载,大于 0 时表示同属于此视频数据对象的前几个负载的数据长度之和
7	08	标志符	
8 ~ 11	7a 0b 00 00	数据对象的长度	单位为 Byte
12 ~ 15	74 5a 88 01	数据对象的时间戳	单位为 ms
16 ~ 17	7f 08	负载长度	单位为 Byte

将 02(82)… 00 00 00 00 08 作为寻找视频数据对象的确认符,找到每一个视频数据对象的起始位置,并记录其数据对

象类型、序号、长度以及时间戳。

2.3.2 找到视频数据对象的所有数据

视频数据对象的所有数据就是其所包含的所有负载中的数据。一个负载的数据如果在当前视频数据包中找不全,那么到紧临的下一个视频数据包中查找其余数据。

若一个视频数据对象只包含一个负载,那么其位置确定后,即可找到所有数据;若一个视频数据对象包含若干个负载,第一个负载位置确定后,还要确定其余负载位置。同属一个视频数据对象的两个负载的头(十六进制)举例如图 6 所示。

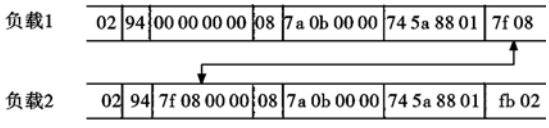

图6 同属一个视频数据对象的两个负载的头示例

可以发现,除了偏移量与负载长度位置上的字符不同外,其他位置的字符都相同,并且当前负载的偏移量等于之前的负载长度之和。由此可见,当第一个负载位置确定后,可以此作为找到其余负载位置的依据。依次将所有的负载位置找到后,依据其长度,可得到负载数据,即可提取到一个完整的视频数据对象。

2.4 封装为 ASF 文件

ASF 文件由三个高层对象组成:头对象(header object)、数据对象(data object)和索引对象(index object)。头对象和数据对象是必需的,头对象必须放在文件的开头部分,数据对象紧跟在头对象之后;索引对象不是必需的[14]。在封装时,只考虑头对象与数据对象。利用 C++ 语言将视频数据封装为 ASF 文件可分为两个步骤:写入文件头;将视频数据组装为包写入。具体步骤如图 7 所示。

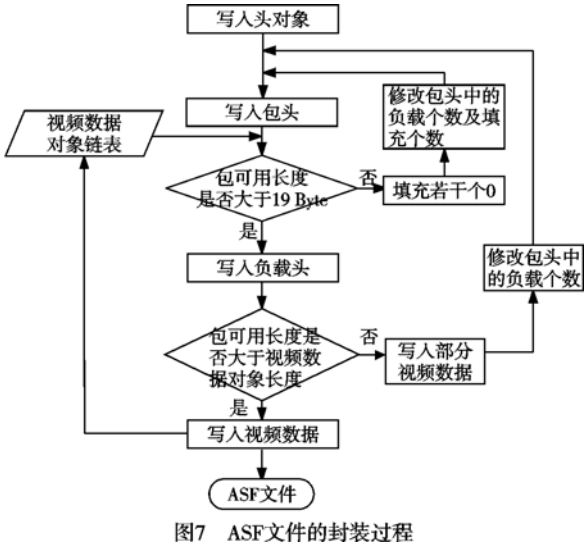

图7 ASF文件的封装过程

2.4.1 写入头对象

头对象包含很多个对象。只选择以下四个必需的对象写入到文件中:文件属性对象(file properties object)、头拓展名对象(header extension object)、流属性对象(stream properties object)、流比特率属性对象(stream bitrate properties object)[14]。

2.4.2 将视频对象数据组装为包

将视频数据对象中的视频数据逐次填充到包中。

1)写入包头 封装文件的包头长度为 13 Byte。举例如图 8 所示。

82 00 00 09 5d	00	00 00 00 00	00 00	89

图8 封装文件的包头示例

各字节的含义如表 8 所示。

表 8 包头各字节的含义

位 置	数 据	含 义
1 ~ 5	82 00 00 09 5d	固定值
6	00	填充字符(padding)0 的个数
7 ~ 10	00 00 00 00	传输时间
11 ~ 12	00 00	持续时间
13	89	8 为固定值;9 表示包内包含的负载个数

在此过程中,需要对填充字符和负载的个数进行修正。

2)写入负载头 封装文件的负载头长度为 19 Byte。举例如图 9 所示。各字节的含义如表 9 所示。

表 9 负载头各字节的含义

位 置	数 据	含 义	备 注
1	02	数据对象类型	02 表示非关键视频对象,82 表示关键视频对象
2	01	数据对象序号	从 1 到 ff 依次排序
3 ~ 6	00 00 00 00	当前负载起始位置相对于所属视频对象起始位置的偏移	单位为 Byte
7	0a	标志符	
8 ~ 11	cb 28 00 00	数据对象的长度	单位为 Byte
12 ~ 15	af 0a 00 00	数据对象的时间戳	单位为 ms
16 ~ 17	28 00	固定值	
18 ~ 19	e0 0e	负载长度	单位为 Byte

因为包的长度是固定的,并且负载中必须包含数据,那么在写入负载头之前,需对包的可用长度进行判别。另外,需对视频数据对象的类型、序号、长度、时间戳以及负载的偏移、长度作出修正。

3)写入视频对象数据 视频对象数据是通过负载承载的。因为包的长度是固定的,所以负载数据的长度受包可用长度的限制:如果包当前的可用长度大于当前视频对象数据的长度,那么承载此视频数据的负载长度就可以设置为视频数据对象长度;反之,则需要把当前视频对象数据拆分为若干个部分,分别写入到不同的负载中去。

依据上述步骤,依次将每个视频数据对象进行封装、写入到文件中,即可封装为一个 ASF 文件。

3 内容异常检测

3.1 视频内容检测

在视频文件播放过程中,可将视频帧保存下来,进行实时台标检测,如判定台标被窜改时则及时报警。台标检测可分为两个步骤:a)台标模板的提取;b)利用图像模板匹配算法进行检测,以判断所播放的视频内容是否正确。限于篇幅,具体做法在这里不再赘述。

3.2 音频内容检测

利用语音转换软件将封装好的音频文件转换为文本文件。收集代表异常内容的词汇、语句,建立异常词库和异常内容语义模式库。首先对文本进行异常词过滤,对其进行预处理和特征提取,进行模式匹配,得到观点倾向性分析结果。步骤如图 10 所示。此部分另行撰文阐述。

(下转第 196 页)

(2007-04-02)[2011-05-07]. http://web.uvic.ca/~hitchner/assign1.pdf.
[4] Mozilla Foundation. XPInstall[EB/OL]. (2010-03-28)[2011-05-07]. https://developer.mozilla.org/en/XPInstall.
[5] BRENT S. XULRunner:a new approach for developing rich Internet applications[J]. IEEE Internet Computing,2007,11(3):67-73.
[6] PARRISH R. An introduction to XPCOM[EB/OL]. (2009-02-16)[2011-05-07]. http://www.ibm.com/developerworks/webservices/library/co-xpcom.html.
[7] Mozilla Foundation. XPCOM[EB/OL]. (2011-01-24)[2011-05-07]. http://www.mozilla.org/projects/xpcom/.
[8] Mozilla Foundation. Introduction to XUL[EB/OL]. (2011-01-24)[2011-05-07]. https://developer.mozilla.org/en/Introduction_to_XUL.
[9] Mozilla Foundation. XBL[EB/OL]. (2009-09-02)[2011-05-07]. https://developer.mozilla.org/en/XBL.
[10] VERDURMEN J. Firefox extension security[D]. Netherlands:Raboud University, 2008.
[11] Security-Assessment.com. CoolPreviews Firefox extension privileged code injection [EB/OL]. (2009-08-25)[2011-05-16]. http://www.security-assessment.com/files/documents/advisory/2009-08-25_CoolPreviews_Firefox_Extension_Security_Advisory.pdf.
[12] SALVATORE G, BENJAMIN L. Gatekeeper: mostly static enforcement of security and reliability policies for JavaScript code[C]//Proc of the 18th USENIX Security Symposium. Berkeley: USENIX Association, 2009:151-168.
[13] SRUTHI B, SAMUEL T K, MARIANNE W, *et al*. VEX: vetting browser extensions for security vulnerabilities[C]//Proc of the 19th USENIX Conference on Security. Ber keley: USENIX Association, 2010:339-354.
[14] MOHAN D, VINOD G. Analyzing information flow in JavaScript-based browser extensions[C]//Proc of Computer Security Applications Conference. Washington DC: IEEE Computer Society, 2009:382-391.
[15] MIKE T L, JIN S L, VENKATAKRISHNAN V. Enhancing Web browser security against malware extensions[J]. Computer Virology, 2008,4(3):179-195.
[16] LDUW M T, LIM J S, VENKATAKRISHNAN V. Extensible Web browser security[C]//Proc of the 4th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Belin:Springer-Verlag, 2007:1-19.
[17] OYSTEIN H, GIOVANNI V. Detecting malicious JavaScript code in Mozilla[C]// Proc of the 10th IEEE International Conference on Engineering of Complex Computer Systems. Washington DC: IEEE Computer Society, 2005:85-94.
[18] LI Zhuo-wei, WANG Xiao-feng, CHOI J. SpyShield:preserving privacy from spy ADD-ons[C]//Proc of the 10th International Conference on Recent Advances in Intrusion Detection. Berlin: Springer-Verlag, 2007: 296-316.
[19] TOWNSEND D. Firefox 3.5 interfaces[EB/OL]. http://www.oxymoronical.com/experiments/xpcomref/applications/Firefox/3.5/interfaces/.
[20] Security-Assessment.com. Yoono Firefox extension privileged code injection[EB/OL]. (2010-01-13)[2011-05-16]. http://www.security-assessment.com/files/documents/advisory/2010-01-13_Yoono_Firefox_Extension_Privileged_Code_Injection.pdf.
[21] Security-Assessment.com. ScribeFire Firefox extension privileged code injection[EB/OL]. (2009-08-24)[2011-05-16]. http://www.security-assessment.com/files/documents/advisory/2009-08-24_ScribeFire_Firefox_Extension_Privileged_Code_Injection.pdf.
[22] Security-Assessment.com. Feed sidebar firefox extension privileged code injection[EB/OL]. (2009-08-24)[2011-05-16]. http://www.security-assessment.com/files/documents/advisory/2009-08-24_Feed_Sidebar_Firefox_Extension_Privileged_Code_Injection.pdf.

(上接第 190 页)

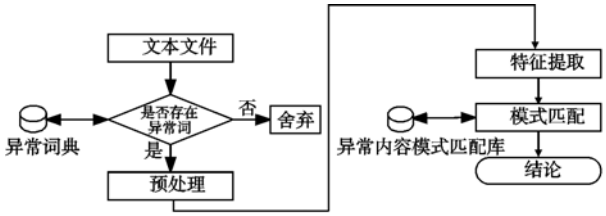

图10　文本异常内容检测

4　结束语

本文以 P2P-TV 内容监管为目的,通过对 P2P-TV 监控系统的介绍,主要实现了将 PPTV 数据流实时还原为 ASF 文件的实现方法。最后对台标及音频转换后的文本进行了检测,达到较好的效果,基本上满足了对 PPTV 进行内容检测的要求,为进一步实现对其他 P2P-TV 应用的内容监管打下了良好基础。

参考文献:

[1] PPTV[EB/OL]. http://www.pptv.com.
[2] PPStream[EB/OL]. http://www.ppstream.com.
[3] UUSee[EB/OL]. http://www.uusee.com.
[4] Sopcast[EB/OL]. http://www.sopcast.com.
[5] QQLive[EB/OL]. http://www.qqlive.com.
[6] HEI Xiao-jun, LIANG Chao, LIANG Jian, *et al*. Insights into PPLive: a measurement study of a large-scale P2P IPTV system[C]//Proc of International World Wide Web Conference. 2006.
[7] WU Chuan. Large-scale peer-to-peer streaming: modeling, measurements, and optimizing solutions[D]. Toronto: University of Toronto, 2008.
[8] 袁雪美,王晖,张鑫,等. P2P 流量识别技术综述[J]. 计算机应用,2009,29(12):11-15.
[9] P2P 录像专家 v2.43 正式版[EB/OL]. http://www.newmediasoft.cn.
[10] PARISH D J, BHARADIA K, LARKUM A, *et al*. Using packet size distribution to identify real-time networked applications[J]. IEE Proceeding-Communications, 2003, 150(4):221-227.
[11] LI Jin, ZHANG Xin, ZUO Xiao-Liang, *et al*. Using packet size distribution to iIdentify P2P-TV traffic[C]//Proc of International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC 2010). 2010.
[12] 张鑫,李进,王晖. 基于频道签名的 P2P-TV 流量精细识别[J]. 计算机工程,2010,36(24):9-14.
[13] 蒋海明,张剑英,刘琼,等. PPLive 协议分析及流量识别[J]. 电视技术, 2009,49(5):21-24.
[14] Microsoft Corporation. Advanced systems format (ASF) specification, revision 01.20.03[R]. 2004.
[15] WU Chuan, LI Bao-chun, ZHAO Shu-qiao. Exploring large-scale peer-to-peer live streaming topologies[J]. ACM Trans on Multimedia Computing, Communications and Applications, 2008,4(3):1-23.