Understanding the Brains and Brawn of Illicit Streaming App

Kong Huang[1], Ke Zhang[1], Jiongyi Chen[2], Menghan Sun[1], Wei Sun[1], Di Tang[1], and Kehuan Zhang[1(✉)]

[1] The Chinese University of Hong Kong, Hong Kong, China
khzhang@ie.cuhk.edu.hk
[2] National University of Defense Technology, Changsha, China

Abstract. Content piracy has been the largest threat to the whole TV and Media Industry around the world causing billions of dollars of economic loss. Copyright-protected contents, including movies, live soccer matches, basketball games, dramas, etc., have been taken by "content pirates" and redistributed through the Internet. In the last few years, illegal IPTV providers in the form of an illicit streaming app running on smartphones, smart TVs, and illicit streaming devices have become extremely popular and take away significant subscription revenue from legitimate pay-TV operators around the world. In this research, we study the illicit streaming ecosystem from a new perspective by looking at the illicit streaming apps: we build a semi-automated forensic tool to analyze the common codes, libraries, and network traffic of the apps that facilitate illicit streaming services. As a result, we are able to investigate their background and identify the technology providers behind them. Our research provides insights into the proliferation of illicit streaming services in app markets, as well as the overall illicit streaming ecosystem.

Keywords: Content piracy · Video streaming · Digital forensics

1 Introduction

Illicit streaming has been a new form of copyright infringement in the last decade. The Intellectual Property Office of the UK government in [40] defines illicit streaming as the watching of content without the copyright owner's permission by any means, not just via hardware devices. This could be using a smart TV, laptop, or mobile phone. Illicit streaming devices (ISD) are physical set-top boxes or USB sticks connected to your TV. The hardware itself is not illicit, the crux of the problem is the illicit streaming app (ISA) running on top. By definition, we need to confirm with each copyright owner if the video contents on a streaming app are unauthorized before they are officially called ISA.

ISA has become the most popular form of streaming piracy with 12 million active users in the US [18]. Sandvine [44] has issued a report in 2017 about the ISD piracy ecosystem, including unlawful device sellers and unlicensed video

P. Gladyshev et al. (Eds.): ICDF2C 2021, LNICST 441, pp. 194–214, 2022.
https://doi.org/10.1007/978-3-031-06365-7_12

providers and video hosts, which brought in revenue of an estimated 840 million a year in North America alone, at a cost of USD 4.2 billion to the entertainment industry. In a report from CNN Money in April 2018 [23], Southeast Asia has become the epicentre of the manufacturing and distribution of ISD which only cost as low as USD 150. In Singapore alone, the pay-TV subscribers have fallen 15% over the past 2 years because of piracy and it is predicted that it shall fall another 15% in the next 2 years.

From the above reports, it is not difficult to see that content piracy through illicit streaming is a huge problem affecting the content industry on a global scale. Law enforcers around the world face huge challenges because there are so many illegal streaming apps in the market [31]. After one ISA has been taken down, a new one just pops up with a different name within weeks if not instantly, which is also known as the so-called "whack-a-mole" phenomenon [24]. Thousands of streaming apps are available in app stores, some apps that have been removed due to their infringing contents but remain available as APKs to be downloaded from other Android app markets, which makes Android a more popular choice of ISA [1].

The intuition behind this study is that the proliferation of ISA, with overlapping contents and non-trivial streaming technology and infrastructure, might have been fueled by the existence of technical enablers of some kind. ISAs that had been shut down would come back easily with another brand and app name. As such, our research aims to find out if ISAs are built with similar libraries, we can further identify the developers and their roles in the illegal streaming ecosystem. To this end, we collected data from two groups of Android streaming apps. The first group are the 1,360 live streaming apps collected from various Android app stores. The other group are the confirmed set with ISAs from different sources, which include ISAs taken down by some countries in the news, the ISAs verified with the content provider during our research, and the default apps found on known ISD in Hong Kong. We shall use the open-source code repository Github [7] or Google search engine to do a background check and identify the developers. We developed a tool to perform automatic software similarity analysis comparing methods, classes, and libraries in the installation package of the current application. The tool could also analyze network traffic and generate reports of selected ISAs during runtime.

Through our analysis of 21 confirmed ISAs and 10 legal streaming apps like YouTube, Netflix, etc., we found a set of third-party libraries that are potentially illicit streaming libraries. We checked the developers' background and investigate their involvement in the content piracy ecosystems. Many developers put their libraries in Github but our case studies show that these are just cover-ups of their larger and perhaps illegal roles behind. For instance, we found a company in Shanghai called Binstream.io which is the technology provider of a library "Tvbus" which are used by two of the confirmed ISAs. They operate an illegal streaming service called "Sopplus TV" and distribute over a hundred apparently unlicensed channels. We have reported to respective owners of the content and they acknowledged our findings. Using software similarity between 21 confirmed ISAs and 1360 apps from Android app stores, we can find another 23 apps that

are similar with over 33% third-party libraries overlap. For instance, based on the ISA which is called UBLive, we can find V-Sat which is 94% and BR.TV which is 88% similar to that ISA. This demonstrates the use of software similarity to automatically identify potential ISAs among thousands of apps in the open app markets.

Contributions. Our work makes the following contributions:

- *New investigation.* We perform the first investigation into the illicit streaming ecosystem by analyzing illicit streaming apps at a large-scale. We developed an semi-automated tool which combines software similarity analysis and network analysis, helping us correlate different ISAs and identify technology developers.
- *New Discovery.* We demonstrated using our case studies of the ISA that common developer of multiple ISAs exist. We showed their involvement in the illicit streaming ecosystem through manual investigation, which may suggest a new choke point in fighting piracy. We shared the results with the Premier League [17] and Asia Video Industry Association (AVIA) [16], both organizations acknowledged our findings and shall take further investigations.

Roadmap. For the rest of the paper, we shall summarize the underlying operations and existing forensics of illicit streaming services as the background of the research in Sect. 2 and then propose our methodology in Sect. 3. In Sect. 4, we shall carry out a large scale study on the streaming app, and analyze the forensic findings using real cases. In Sect. 5, we shall list the related works, limitations, and future developments, and in Sect. 6 the conclusion.

2 Background

Free and Subscription Based Illicit Streaming Service. There have been two types of illicit streaming services. The ad-supported free illicit streaming service and the subscription-based illicit streaming service. The ad-supported service focus on drawing traffic and display digital advertisement to visitors. They can use cyberlockers, cloud storage, and BitTorrent as content distribution intermediaries. The subscription-based illicit streaming service collects money directly from the consumer, say for USD 300 per year with ISD, providing hundreds of high-quality live channels. Figure 1 illustrates the setup of ISA on mobile devices and TV set-top boxes (a.k.a. ISD). First, unlicensed contents are put on the CDN (Content Delivery Network) [25]. There are hundreds of stable and high-quality live channels, e.g. HBO, Disney, BBC, CNN, etc. The illegal service provider will use large Cloud and CDN providers for hosting and distributing content. At the client-side, ISA developed on smartphones or ISD will connect to the cloud server for control signals and stream content from the CDN. These illegal service provider may have their own illegal content sources or simply get from unlicensed content providers [1].

Current Measures to Combat ISA. International media companies and large pay TV operators will employ content protection solution providers (e.g. Friend MTS [38], OpSec [41], etc.) to investigate, monitor, and take-down illegal streaming services. Since there are thousands of ISA in the market, the content providers will select a handful of the most popular apps for monitoring. Network forensics will be performed to identify the underlying IP addresses of video streams for issuing blocking orders to internet service providers by the administration or court. The solution would cost as much as a hundred thousand US dollars per year. There are several inefficiencies in the current measures: (1) only a handful of most popular apps can be investigated in each cycle, (2) there is no simple way to find out which streaming apps are more likely to be illicit, (3) there is no knowledge base of confirmed ISA to assist the future investigation of new streaming apps.

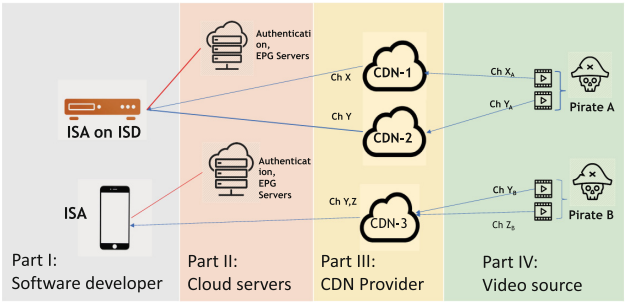

Fig. 1. Common ISA architecture

Ethical Discussion. To abide by an ethical framework throughout this study, when making ISD purchase, we limited to the top models in Hong Kong [27]. In total, we spent no more than USD 1,000 to minimize the amount of money flowing to the illegal services. We justify that by detailed analysis for various ISD (in fact it is the analysis of the default ISA running on top), which could not be performed without the purchase. We have sent our respective findings to the Premier League [17], AVIA [16], MoPub [9], AdColony [3], and Adincube [10]. It is also important to note that although this work may present information that could point to specific kinds of actions, including but not limited to take-down notices. It should not be used in any sort of legal proceedings as-is.

3 Our Methodology

ISA is one of the brains and brawn of the content piracy ecosystem. It made access to illegal content extremely easy to the mass audience. In this research, we aim to look at how the ISA are built and who are the developers behind them. We first leverage program analysis techniques to identify illicit streaming libraries from a known set of confirmed ISA. Then we perform large-scale analysis

to identify other potential ISA that shares the same or similar illicit streaming libraries. This could provide some insights on how the potential ISA is similar to confirmed ISA by the illicit streaming libraries that they share.

In Fig. 2, we illustrated the overall workflow of our methodology. The first step is to crawl and download Android Application Package (APK) files of streaming apps from the common Android app store (1). Then manual upload confirmed ISAs and legal streaming apps to the system (2). The APK files will be decompiled into method objects (3). Each APK file will generate a method objects set and the set will be hashed using the MD5 function to build a feature vector for each APK file. When all APK feature vectors are completed, a feature matrix is generated in (4). Third-party libraries will be detected in (5) and the similarity between each pair of ISA is computed by Jaccard similarity in (6). Shortlisted APK (7) is connected to the internet via a VPN server. All IP traffics including requests (8) and responses (9) will go through the VPN server and be saved as a PCAP file [49]. The PCAP file is analyzed in (10). The overall result is a JSON file (11).

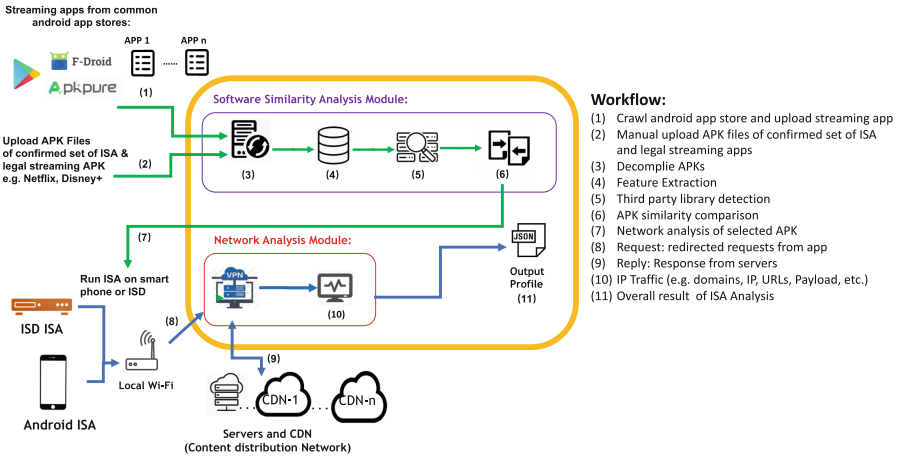

Fig. 2. Workflow of the ISA analysis

3.1 Detection of Illicit Streaming Libraries and Potential ISAs

In this subsection, we describe the automated techniques to find out illicit streaming libraries as well as potential ISA. However, the vastness of the ecosystem, the sheer volume of illegal streaming URLs, and the professionalization of pirate operations made it overwhelming to efficiently study ISA [22].

App Collection. To automate the search and crawling of apps from multiple app stores, we crawl .apk files from a list of popular Android app stores [2] [12] (the full list can be found in Sect. 4.1) using an open-source APK crawler [37].

Decompilation and Representation. The initial workflow of .apk files analysis consists of the following steps: 1) decompile each input APK adapting Androguard

[19] to get the Control Flow Graph (CFG) of each method and the methods list. 2) The opcodes in each CFG are hashed using MD5 [48] in depth-first order to bypass the potential name obfuscation and generate the signature of each method.

Feature Extraction. The process of feature extraction of Android APK in different level is inspired by LibD [34]. LibD focuses on clustering and identifying the third party libraries, while our target is to measure the similarity of a given pair of APK by observing the overlapped methods among Apps. Thus, we compare the similarity of APK pair in method level to retrieve as much as possible underlying similarity instead of merely library identification. The features can be extracted from the Control-Flow-Graph of each method by computing its MD5 hash code of the concatenation of opcodes. The full process is shown in Algorithm 1.

Algorithm 1: APK similarity comparison by method-level signature.

Data: A, ISA, OSA
Result: $simi\_scores$
Function `SignatureExtraction(`a`):`
  $methods_a$ = GetMethods(Decompile(a))
  $methods\_opcode\_str_a = \emptyset$ `//Initialize an empty set`
  for *method in* $methods_a$ **do**
    CFG_{method} = GetCFG($method$)
    $opcode\_str_{method}$ = GetOpcodeStr(CFG_{method}))
    $methods\_opcode\_str_a$ = cat($methods\_opcode\_str_a$, $opcode\_str_{method}$)
  end
  sig_a = Hash($methods\_opcode\_str_a$)
return sig_a
Function `SignatureOSA(`OSA`):`
  $sig_{OSA} = \emptyset$ `//Initialize an empty set`
  for *osa in* OSA **do**
    $sig_{OSA} = sig_{OSA} \cup$ `SignatureExtraction(`osa`)`
  end
return sig_{OSA}
Function `Similarity(`A, ISA, OSA`):`
  sig_{OSA} = `SignatureOSA(`OSA`)`
  $simi\_scores = \varnothing$ `//Initialize an empty table`
  for a *in* A **do**
    for isa *in* ISA **do**
      sig_{isa} = `SignatureExtraction(`isa`)`
      sig_a = `SignatureExtraction(`a`)`
      $simi\_score_{<a,isa>} = \frac{|(sig_{isa} - sig_{OSA}) \cap sig_a|}{|sig_{isa}|}$
      $simi\_scores[a][isa] = simi\_score_{<a,isa>}$
    end
  end
return $simi\_scores$

APK-Level Similarity Comparison. Each APK is represented by a set of method signatures that represent the features extracted from the above step. The reason for using the method-level signature instead of higher syntax structures is to retrieve the underlying overlapping as much as possible. The underlying methods used by the legal video applications will be excluded and the remaining methods set of the streaming app will be used to compare with the given APK, to identify potential ISAs. To eliminate the libraries that are also shared by legal streaming apps, we build a "standard streaming libraries" set of libraries used by well-known and most downloaded streaming apps in the US and China. The apk files of these official apps are collected from the app stores and analyzed using the same approach as the ISAs to get the full library list and methods-level signatures. The reason we included China streaming apps is that many ISAs in our confirmed set comes from ISD made in China and it helps to improve the accuracy of the illicit streaming libraries overall. The legal apps that we used in generating the standard streaming libraries are YouTube, Netflix, Disney +, Hulu, and HBO Max from the US. And iQiyi, Tencent video, TikTok, Youku, and Mango TV from China. The result set of unique libraries are the illicit streaming libraries.

Potential ISA Libraries Extraction. We extracted the library list of all apps in our dataset including the confirmed 21 ISAs by adapting LibD [34]. Most library names can be retrieved and in plain-text, while the rest part of libraries may be obfuscated during compilation to hide their names and shown as meaningless placeholders such as "a/b/c/". Though the library names obfuscated, the underlying methods level signature remains unchanged.

3.2 Network Traffic Analysis

Network Traffic Capture. For ISA that can run on ISD, we would not be able to use remote control APIs like the case in Roku or Amazon Fire TV [36], so we built a simple ISD control robot with an infra-red output signal of "channel up" after the ISA is launched manually. The robot will scan through all the channels available on the ISD with 15 s intervals in between. As to mobile ISA, due to different user interfaces, registration, and subscription required, we need to manually navigate the selected ISA. To perform a large scale network analysis of Mobile ISA, we need automatic Android app navigation. It is technically feasible and we shall discuss that in Sect. 5.3.

Table 1. Table of notation of Algorithm 1

Notation	Definition
A	The collected APK dataset
$a \in A$	The a is an input apk file in dataset A
isa, ISA	The set of confirmed ISA files
osa, OSA	The set of official (legal) streaming apps (OSA)
Decompile()	Decompile the input APK file a
$methods_a$	A set with all methods of APK a
GetMethods(a)	Generate methods of decompiled a
$method$	A method in $methods_a$
GetCFG($method$)	Generate the control-flow-graph of $method$
CFG_{method}	The control-flow-graph of $method$
GetOpcodeStr()	Generate the opcode string of the CFG of methods
$opcode\_str_{method}$	The opcode string list of $method$
$cat(str_1, str_2)$	Concatenate the string1, str_1, and string2, str_2
$methods\_opcode\_str_a$	The opcode string list of all methods of APK a
Hash()	MD5 hash function
sig_a, sig_A	The signature of a or A computed from hash function
$simi\_score_{<a,isa>}$	Similarity comparison between a and isa
$simi\_scores$	A key-value table for storing APK pairs and their similarity

Network Traffic Analysis. The network data collected during the startup of the apps and the content streaming. Our tool could create interaction diagrams and extract the IP addresses, domains, name of hosting companies, CDN, and their autonomous system network number automatically. We also look into the text strings from HTTP requests (e.g. Streaming URLs) and responses (payloads) and see if there is any match with the library or developer names we found in the software similarity module.

4 Findings

4.1 Data Collection

Streaming Apps from App Stores. We selected 34 common Android App stores, e.g. APKure, APKMirror, AppBrain, Aptoide, F-Droid, etc. We search these app stores with a combination of "live", "tv", "iptv" and "streaming". A total of 1,360 apps in Fig. 2 are uploaded for features extraction. Such automation ensures we could cover as many apps as we want to meet the challenge of the vast volume of streaming apps in app stores.

Confirmed Set of ISAs. First, we collected the default ISAs from ISD. In May 2018, the Customs and Excise Department of Hong Kong has seized four brands of ISDs (Boss V2, EVPad 3, Magic Plus, and UBox ProS) through a series of raid of retail outlets in Hong Kong [27]. Between April and July 2019, we purchased these four ISDs from the Hong Kong market plus another ISD recommended by the same store called the TurboTV S box. From them, we could download a total of 9 ISAs. Secondly, we searched the Internet and found from different sources about the shutdown of another 7 ISAs. And finally, we contacted a content owner's office in Singapore and they helped verified 5 more ISAs. Therefore a total of 21 ISAs has been included in our confirmed set of ISAs in Fig.2.

Software Analysis. We decompile 1,360 APK files and confirmed set with 21 ISAs using Androguard [19] to get the list of all methods and classes from all APK files. Meanwhile, the constant string variables found during decompilation are also extracted from the APK files for further online search analysis. Then, the control flow graph (CFG) of each method is built using the opcodes of smali code corresponding to each method to overcome the name obfuscation. This is inspired by the previous work of LibD [34]. After that, the CFG of all methods for each APK is hashed into a 16 bytes string as the signature of each method using MD5 [48] and consist of a feature vector. Note that here we only construct method-level features so the order of methods or elements in feature vectors are ignored. The similarity of a pair of feature vectors can be computed by Jaccard Similarity [46].

4.2 Key Findings on Illicit Streaming Libraries

There are total of 3,299 named libraries found from 1,360 APK, we stripe out standard libraries common between 10 legal streaming apps and focus on the use of illicit streaming libraries. Here are the key findings:

– **Online Advertising Platform.** The top matched library is MoPub used by 221 apps. MoPub is a mobile advertising platform owned by Twitter [9], which help the apps to monetize the traffic they gain from offering free illegal channels. This is the major business model for free illicit streaming services [1]. Not only MoPub, in the same category, there are also 83 apps using Adcolony [3], 19 using AppLovin [5], 10 using AdinCube [10], 7 using Vungle [14], 5 using Avocarrot [8] as shown in the Table 3.
Implication: MoPub has established a policy for publishing partners and copyright infringement contents are prohibited. But in our results, they are inside one confirmed ISA "Terrarium TV". As for AdColony, they are awarded the Certified Against Fraud seal by the Trustworthy Accountability Group, but they are used by 4 ISA, "Terrarium TV, Tea TV, UKTV Now, LiveNet TV" [4]. The issue lies in the enforcement of the company policy and our tools

could help identify potential ISA using legitimate advertising platforms. Once the content owner confirms respective content infringement in the ISA, the advertising platform could immediately take action to stop serving ads and hence cut out the money chain to the ISA. It is a win-win situation for the content owner and advertising platform to ensure the later are not aiding and abetting ISA like TeaTV, UKTVNow and Terrarium TV [1].

- **Video Player.** Another major common library we found is the player or media framework. There are (1) "Vitamio" developed by Yixia.com from China [13] in 22 apps including 1 ISA "Supersport", (2) TVBus developed by Binstream.io from China in 4 apps including 2 ISA "SunshineTV, Supersport", and (3) NagasoftPlayer developed by Nagasoft from China in 3 apps including 2 ISA "SunshineTV, Supersport", (4) Forstwire from Forstwire Project in 2 ISA "MagicTV and ShowboxTV", and (5) Gemini from 2 ISA "SunshineTV, Supersport". Details are shown in Table 3.
  Implication: With the developer information, we can do a background check and see if the software companies have legitimate business registration and operations. Or if they are in fact part of the content piracy ecosystem fueling the development of multiple ISA. Which we shall discuss in detail in Sect. 4.5 Case Studies.
- **Data Provider.** There are also TV shows and movie data API provider called Trakt.tv, and the library uwetrottmann/trakt5 has been used by 4 apps including three ISAs (Terrarium TV, Magic TV, and TVZion). Trakt.tv API powers thousands of apps like media center plugins, mobile apps, watch apps, command-line utilities, and smart home integration.
  Implication: The free to use data API get quality data from legal services like Netflix, Hulu, and Disney+ and helps create many useful apps for good. However, it is also aiding and abetting many ISAs in the market.

4.3 Key Findings on Detection of Potential ISA

For each of the confirmed ISA, we shall strip off the common libraries from Android, Google, or other benign standard Java libraries, and check its similarity with all the 1,360 streaming app. We found 64 unique streaming apps that are similar to the one or many of the ISAs in the confirmed set. These 23 apps have over 33% similarity to one or more ISAs. We show the result in Table 2. Some of the similar apps to the confirmed set of ISAs are detailed below:

- **UBLive, UBVod and similar apps.** UBLive and UBVod are default ISAs found in ISD called UBox and are actually identical. V-Sat is 94% and BR.TV is 88% similar to them, which means these 2 apps are built nearly the same way as UBLive and UBVod. In one of the Android app store, V-SAT is described as "an app developed by Uldeck that brings together around 3,700 channels from all over the world, both general and specialized channels". From the description of BR.TV app in the app store, "With Brasil TV Live, you can watch all Brazilian TV live for free from your smartphone." Both are typical ISA type of promotions [15]. **Implication:** Using our software

similarity comparison between APK, we are able to identify potential ISAs which are developed the same way as the known and confirmed set of ISAs. This helps the content providers to sort out the apps they want to take down first.

– **Tea TV and similar apps.** BN Live TV is 68%, USTV Hub is 56% similar to Tea TV. In one of the app store, BN Live TV has descriptions saying "Enjoy the most popular TV channels in Bangladesh" and USTV Hub has a website called USTVHub.com and describes its service as "Watch live tv channels/shows on your android phones or boxes free of charge", both are typical ISA types of promotions [15]. **Implication:** BN Live TV and USTV Hub may not belong to the same ISA family as Tea TV, but they are over 50% similar, an indication of similarity in functionalities, business models, and data sources. It allows content owners to make an informed decision among thousands of apps in the open market, which one to look into first.

4.4 Network Analysis Result

- Understanding the server IP and CDN provider of ISA. Both Supersport and SunShine TV uses CloudFlare (AS 13335). SuperSport app uses Ensu as CDN (ASN 18978) and SunShine TV uses OVH as CDN (ASN 16276). From our software analysis, we know the two ISA are 91% similar, yet they are using different CDN network providers because they are accessing a different list of contents.
  Implication: The software similarity analysis provides lower-level information that network-level analysis cannot provide.
- Call flow analysis. Our tool will automatically generate an interaction diagram for the ISA traffic captured. In Fig. 3, the Sunshine TV ISA will contact "weixingdianxitv.com" and in the HTTP request, it contains a phase called the "gemini-iptv" which are also resonant with the name of illicit streaming libraries that Sunshine TV used, e.g. com/gemini/application, com/gemini/-play, com/gemini/turbo, etc.
  Implication: The network analysis of apps runtime data may sometimes support the results of our static software analysis.

4.5 Case Studies

To further understand the effectiveness of our methodology in finding common infrastructures behind illicit streaming ecosystem, and developers supporting it, we analyze each case in-depth and share the results in the following.

TVBus and Binstream. Within the confirmed set with 21 ISAs, Sunshine TV and SuperSport contain the same library called "tvbus". We searched Github using the name "tvbus" and found that it is developed by a company called "binstream.io" (https://github.com/binstreamio/tvbus.android). This suggests that

the two ISA share the common library "tvbus" developed by Binstream.io. Binstream.io is a P2P technology solution company in Shanghai city. On their website http://www.binstream.io, we learned that "tvbus.android" is a live streaming android SDK based on "P2P technology", and it's available on Android, Win32, macOS and Linux platform. We also found their media distribution solution called "SopPlus TV" using Binstream technology which matched the name of the content server in Github through tvbus protocol. We crawl and scrape the information from Binstream.com and got a channel list in JSON file including 108 channels distributed from the server of Sopplus TV, including HBO, CNN, BBC, and NHK which are unlikely to be licensed to a software company in China.

To relate our findings to the physical world, we extend our investigation manually, we contacted the company binstream.io and successfully got a demo APK named P2Player from them to evaluate their solution. The P2Player app comes with a dozen of live channels for testing purpose, connecting to a server also called "SopPlus TV".

From the above evidence, we could see that the company Binstream.io not only supplies its P2P-CDN hybrid technology solution in the form of a software development kit (i.e., TVBus library), they are involved in some sort of operation of an illegal streaming service called the "SopPlus TV", with unlicensed international live channels.

In summary, we found that Binstream.io is playing multiple roles in the illicit streaming ecosystem. Firstly as a technology provider supporting many ISA and ISD through "TvBus". Secondly, it operates an illegal streaming service in the name of "Sopplus TV". Thirdly, through our contact with them, they served as an illegal re-seller of over a hundred unlicensed live channels. We notice that the site binstream.io could not be accessed recently and for completeness sake, we include the site's image in Feb 2020 through web.archieve.org as reference [6].

Gemini IPTV. Sunshine TV and Supersport apps are default apps from Turbo TV S ISD. Besides "tvbus" they shared a set of libraries from "Gemini", called Gemini.custom, Gemini.play, Gemini.base64, Gemini.pay. While Sunshine TV has one more Gemini.Turbo library which Supersport did not. On the contrary, Supersport app has Gemini.ssport library which Sunshine TV did not. This suggests that both apps are more likely built with the same set of libraries. From the interaction diagram generate by network analysis of Sunshine TV Fig. 3, it contains a phase called the "gemini-iptv" in its content request, such result from network analysis resonant with the software analysis.

We further looked into the "Gemini" libraries using the Google search engine and found a relevant site called the geminiproject.tv. On this website, it is advertised that you can get over 22,000 channels from all over the world and they also offer a free trial and support multiple platforms. We tried to contact the service provider but the email is bounced and no response from their official Skype account. There is a Smart IPTV app available for download from the same website. We search further and find that the Smart IPTV service is hosted at

the siptv.app website, while its subscription is sold on another website called IPTV.shop. From which one can subscribe 3-months illegal service at 30 Euros. Someone could also sign up to be a distributor of "Smart IPTV" services. In [42], Pandey et al. described "a middle-man third-party service, Omniverse, which offers most of the infrastructure and content required to operate a streaming media service". The distributors clearly stated the support from Omniverse in their websites and it was easy to establish the case of content infringement and took legal action. However, in our findings, the Sunshine ISA (and Supersport ISA) on TurboTV S ISD are operated as separate apps streaming from different CDN, which looks unrelated in their operations. Through the software analysis, we could associate different ISA in the market and identify more directions for further investigation.

5 Related Work and Discussions

In this section, we review prior researches about illicit streaming services and mobile app similarity and discuss about future works.

5.1 Illicit Streaming Services

There are some previous works on free illegal streaming services, their security, and user privacy issues. Rafique et al. [43] presented the first empirical study of free live streaming services, analyzed the deceptive advertising content within those free websites, and developed a classifier for potential dangerous web pages. Ibosiola el al. [29] studied the use of cyberlocker for online video piracy and identified a number of communities based on shared domains, shared hosting facilities, and high-level HTML similarity. Hsiao el al. [28] built a system to automatically detect new links to illegal live sport streaming sites and provided evidence that these sites seek to track and identify users extensively. In summary, previous researches utilized web crawling tools, HTML page similarity, and HTML features to automatically classify malicious web pages. In this paper, we look beyond websites and target a new and more popular form of content piracy using streaming apps on smartphones or streaming devices. The automated tools required to analyze web pages are different from the one we build to identify common libraries among ISA. To our best knowledge, this is the first study on software similarity of ISA.

There are a few researches on subscription-based illicit streaming services. Nikas et al. studied Kodi app and Kodi add-ons in [39]. Since the crackdown of Kodi by authorities around the world in 2018 [42], different Android ISA has gained more traction. In [42], Pandey et al. presented the first analysis of infringing subscription-based IPTV services and measured the network infrastructure, third party software providers, payment, and other intermediary services that

are used by a subset of the infringing IPTV ecosystem. Inspired by the methodology in [42], we found that an important part of the ecosystem is missing, which is the client-side ISA and their developer. This is important as we have shown in our case study, technology solution provider or developer of ISA is not only taking the development role. In fact, some of them are performing like an "integrator" role, putting together pirated content sources, the platform, and the ISA for end users. They guarantee the ISA's service quality, time-to-market, stability, and scalability of the illicit streaming service.

In summary, previous works focused on network analysis, web crawling or URL link discoveries, which are limited by the pre-requisite to subscribe to different illegal services before they can analyze the service [42]. Our work help to fill in the void of studying the ISA itself and the developer ecosystem behind it. It provides a large-scale analysis and continuous screening to find potential ISA in Android app store, using the similarity to the illicit streaming libraries from known ISA. It is a simple and cost-effective method bypassing the complexity of service subscriptions and specific navigation of different apps.

5.2 Mobile Application Similarity

Our software similarity comparison approach mainly focuses on the syntax-based feature extraction of control flow graph in method level of Andriod APK, to find as many overlapping as possible. We use the Jaccard Similarity to measure the similarity among many ISA. Note that the features, e.g. values hashed from methods represented by Control Flow Graph (CFG), extracted by our approach are essentially from the program syntax. Syntactic features are straightforward representations of different program units of the target APK files. Similar methods have been widely used by many existing works for program similarity comparison and code clone detection [30,32] Besides syntax-based similarity method, there is also the semantic-based similarity method [30,33]. Ideally, they retrieve features by modeling the functionality of the program and usually reveal the underlying similarities of code snippets more accurately [26,35,45]. Similar to common syntax-based similarity methods, they are more suitable for general-purpose problems. We leave it as future works to integrate some suitable semantics-based methods to improve the accuracy of similarity between apps. We are also aware of the methods to prevent reverse engineering of Android applications, for example, dynamic code modification, dynamic loading, anti-debugging, etc. [50]. The challenge is well resolved by known techniques and tools like DexHunter [50], de-obfuscator [21] and Anti-ProGuard [20]. Our tool could also be integrated with these tools in the future. Last but not least, we focus on Java language because the Android APK is usually written in Java source code. We would consider supporting Kotlin [47] language as part of the future work.

5.3 Limitations and Future Developments

Firstly, our methodology could easily extend to Apple iOS applications using different software analysis tools. Secondly, we have focused on English based Android app stores and Github in this paper. In our research, we found that there are 21 apps using Vitamio including colombia-tv, live-tv-arabic-v11, live-tv-chile-chile2, live-tv-peru-per, live-tv-philippines, pak-india-live-tv, turkish-live-tv, and venezuela-ver-tv. It shows that ISAs could be developed for different countries and some of them may only be found in country-specific search engine and app stores around the world. By applying our analysis to more app stores, international ISA database could be built. Last but not least, spreading malware has been identified as one of the revenue streams for illicit streaming platforms [1]. Given the large library database we have accumulated, we could add the malware scoring system [11] to identify malicious apps.

6 Conclusion

In this research, we focus on specific digital forensic challenges of analysing Android streaming apps in content piracy ecosystem. We propose an additional software analysis before the more resource intensive network analysis and human investigation. We developed a tool to capture and analyze the APK of 1,360 apps and used that to explore the hypothesis of a common ISA developer supporting multiple illegal streaming services. The result is positive in that (1) it automates the software analysis of ISA to identify illicit streaming libraries and their usage in more streaming apps and (2) it helps identify more streaming apps in the Android markets that are similar to the ISA identified. In the case studies, we are able to identify common developer behind ISAs called the "Binstream.io" and "Gemini", and report that to content owners who acknowledged our findings. It proves that common ISA developers exist and act as an active part of the illicit streaming ecosystem. We also identify 23 similar apps using 21 confirmed ISAs. This demonstrates a new method in tackling content piracy through large-scale and automatic software similarity analysis in the identification of potential illicit streaming apps.

Acknowledgement. We thank anonymous reviewers for their insightful comments. This work was supported in part by National Key Research and Development Project (Grant No. 2019YFB1804400), and Hong Kong S.A.R. Research Grants Council (RGC) General Research Fund No. 14208019.

Appendix:

Table 2. List of streaming apps similar to confirmed set with 21 ISAs. Sample of the ISA APK files can be found at https://www.dropbox.com/sh/w6f7ecmbbl05snm/AACUlvuC0atGpJXBwB9Qwyhsa?dl=0.

ISA	No. of similar App	(App name, similarity to ISA)
LiveNet	2	('ustvhub', 0.508), ('bn-live-tv', 0.41309090909090906)
BeeTV	1	('fftv', 0.35528837776616135)
UKTVNow	3	('Embratoria', 0.49648924122310306), ('freeflix', 0.46266515666289165), ('fftv', 0.33620234050585124)
TeaTV	4	('bn-live-tv', 0.6805655403786245), ('ustvhub', 0.5563144021087947), ('Liveflix', 0.40750059908938413), ('Listas Wiseplay', 0.335849508746705)
MagicTV	2	('cinemaapk', 0.34298181818181817), ('BibhuTV', 0.3389090909090909)
Exodus	4	('PatoPlayer', 0.4994514536478332), ('Lots-Sports', 0.3959133296763577), ('Tvbox', 0.35161821173889196), ('live-br-tv', 0.33543609434997257)
UBLive and UBVod		('v-sat', 0.9411764705882353), ('BR.TV', 0.8823529411764706)
Mlive	6	('Mobizen Screen Recorder', 0.40213467273583425), ('PuraTV', 0.40087898289122587), ('jaz-live-tv', 0.3936587662847277), ('visionplus', 0.3738816512321457), ('HD_STREAMZ', 0.3690158530842882), ('Peladculas Online Gratis', 0.36634751216449535)
MBOX	3	('fftv', 0.4028169014084507), ('Embratoria', 0.35975855130784706), ('sportsangel', 0.32756539235412474)

Table 3. List of illicit streaming libraries and apps using them

No.	ISA Library	Developer	Confirmed set of ISA using the library	Streaming App List using the ibrary (Top 10 samples)
1	com/tvbus	binstream.io	SunshineTV, SuperSport	BoxStarFireLive, Sopplus
2	com/gemini		SunshineTV, SuperSport	Nil
3	io/vov/vitamio	Yixia.com	SunshineTV, SuperSport	colombia-tv com-tv-splivetv banana.six live-plus-tv live-sports-tv live-tv-arabic live-tv-chile-chile live-tv-free, live-tv-peru live-tv-philippines pak-india-live-tv playbox sport-live-tv turkish-live-tv venezuela-tv
4	com/nagasoft/player	Nagasoft	SunshineTV, SuperSport	BoxStarFireLive
5	com/frostwire	Frostwire	MagicTV, ShowboxTV	Nil
6	com/mopub/common	MoPub, Twitter	TerrariumTV	free-live-tv-for-india, jocker-iptv, listas-iptv-free, underground-iptv, usa-live-tv-channels-free, smart-iptv, nova-era-iptv, cobra-iptv, arabs-iptv, iptv-extreme
7	com/adcolony	AdColony	LiveNetTV, TeaTV, RedBoxTV, UKTVNow, TerrariumTV	Liveflix, TVEspaña, app673606, bn-live-tv, megatvplayer, elMubashir, Fftv, freeflix, tz-hd-live-tv
8	com/applovin	Applovin	BeeTV, LiveNetTV, TeaTV	GSE_SMART_IPTV, Listas_Wiseplay, Liveflix, Liveplanettv, TVTAP, USA Free Live TV, aostv, bn-live-tv, e-DoctorIPTV, freeflix
9	com/avocarrot	Glispa	LiveNetTV, UKTVNow, TerrariumTV	megatvplayer, splive_player
10	com/adincube	Ogury	TerrariumTV, TeaTV, UKTVNow	Listas Wiseplay, Liveflix, bn-live-tv, freeflix, ustvhub
11	com/vungle	Vungle	TerrariumTV, UKTVNow	Embratoria, USTV_Stable, tz-hd-live, uktv
12	com/uwetrottmann/trakt5	Trakt.tv	MagicTV, TVZion, TerrariumTV	CinemaAPK

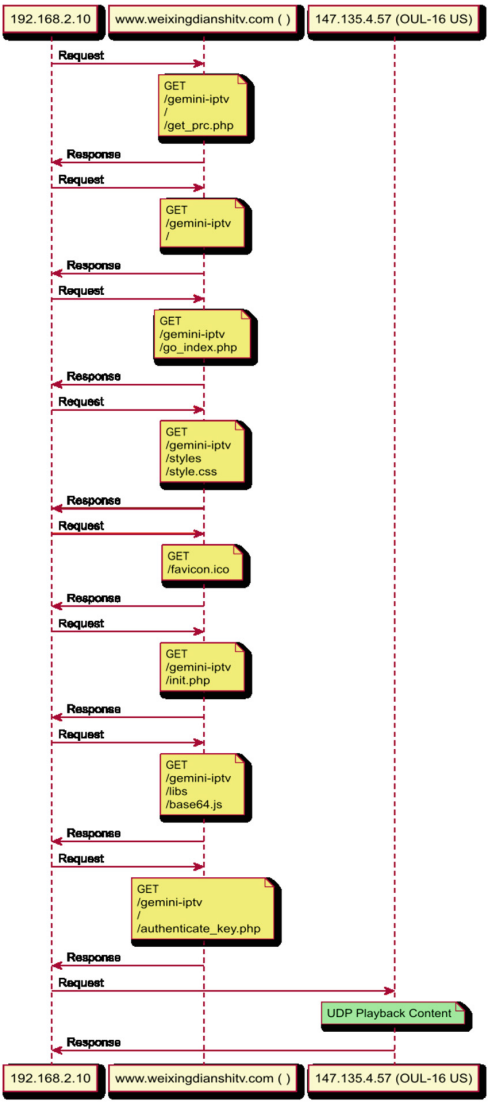

Fig. 3. Interaction diagram of Sunshine TV ISA.

References

1. Illegal IPTV in the European Union (2019). https://www.europol.europa.eu/sites/default/files/documents/tb0319764enn.en_.pdf
2. Ultimate Mobile App Stores List (2019). https://buildfire.com/mobile-app-stores-list/
3. Adcolony (2020). https://www.adcolony.com

4. AdColony receives TAG Certified against fraud recertification for third year running (2020). https://www.globenewswire.com/news-release/2020/03/12/1999716/0/en/AdColony-Receives-TAG-Certified-Against-Fraud-Recertification-for-Third-Year-Running.html
5. Applovin (2020). https://www.applovin.com
6. Archive for binstream.io by web.archive.org (2020). https://web.archive.org/web/20181214170646/www.binstream.io/
7. Github (2020). https://github.com
8. Glispa Connect (Formerly Avocarrot) (2020). https://www.glispa.com/connect/
9. Mopub (2020). https://www.mopub.com/en
10. Ogury acquires AdinCube (2020). https://ogury.com/blog/ogury-acquires-adincube-faq/
11. Quark-Engine (2020). https://github.com/quark-engine/quark-engine
12. Top Best Independent App Stores For Free Apps [2020] (2020). https://medium.com/codixlab/top-best-independent-app-stores-for-free-apps-2020-adf25bc55fd3
13. Vitamio by yixia.com (2020). https://github.com/yixia/VitamioBundle
14. Vungle (2020). https://vungle.com
15. What is illicit streaming? (2020). https://thepcdoc.co.uk/illicit-streaming/
16. Asia Video Industry Association (2021). https://avia.org/
17. Premier League (2021). https://www.premierleague.com/
18. Alliance, D.C.: Boxed in: Hackers targeting piracy devices and apps to infect users with malware (2019). https://www.digitalcitizensalliance.org/news/press-releases-2019/boxed-in-hackers-targeting-piracy-devices-and-apps-to-infect-users-with-malware-report-finds-copy/
19. Androguard_Team: A full python tool to play with android files. https://github.com/androguard/androguard
20. Baumann, R., Protsenko, M., Müller, T.: Anti-ProGuard. In: Proceedings of the 4th Workshop on Security in Highly Connected IT Systems - SHCIS 2017. ACM Press (2017). https://doi.org/10.1145/3099012.3099020, https://doi.org/10.1145/3099012.3099020
21. Bichsel, B., Raychev, V., Tsankov, P., Vechev, M.: Statistical deobfuscation of android applications. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 343–355. ACM (2016)
22. Bushnell, H.: Inside the complex world of illegal sports streaming. Muso Magazine (2019). https://www.muso.com/magazine/inside-the-complex-world-of-illegal-sports-streaming/
23. CNN: Why Fox and Disney hate Singapore's little black boxes (2018). https://money.cnn.com/2018/04/05/media/singapore-piracy-black-boxes/index.html
24. Mostert, F., Lambert, J.: Study on IP enforcement measures, especially anti-piracy measures in the digital environment (2019). https://www.wipo.int/edocs/mdocs/enforcement/en/wipo_ace_14/wipo__4_7-annex1.pdf
25. FACT: cracking down on digital piracy report (2017). https://www.fact-uk.org.uk/files/2017/09/Cracking-Down-on-Digital-Piracy-Report-Sept-2017.pdf
26. Hanna, S., Huang, L., Wu, E., Li, S., Chen, C., Song, D.: Juxtapp: a scalable system for detecting code reuse among android applications. In: Flegel, U., Markatos, E., Robertson, W. (eds.) DIMVA 2012. LNCS, vol. 7591, pp. 62–81. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37300-8_4
27. Hawkes, R.: Eight arrests in HK content piracy crack down (2018). https://www.rapidtvnews.com/2018052952282/eight-arrests-in-hk-content-piracy-crack-down.html

28. Hsiao, L., Ayers, H.: The price of free illegal live streaming services (2019). https://doi.org/10.48550/arXiv.1901.00579
29. Ibosiola, D., Steer, B., García-Recuero, Á., Stringhini, G., Uhlig, S., Tyson, G.: Movie pirates of the caribbean: exploring illegal streaming cyberlockers. In: ICWSM (2018)
30. Jiang, L., Misherghi, G., Su, Z., Glondu, S.: DECKARD: scalable and accurate tree-based detection of code clones. In: 29th International Conference on Software Engineering (ICSE 2007). IEEE, May 2007. https://doi.org/10.1109/icse.2007.30, https://doi.org/10.1109/icse.2007.30
31. Johnson, R., Kiourtis, N., Stavrou, A., Sritapan, V.: Analysis of content copyright infringement in mobile application markets. In: 2015 APWG Symposium on Electronic Crime Research (eCrime). IEEE, May 2015. https://doi.org/10.1109/ecrime.2015.7120798, https://doi.org/10.1109/ecrime.2015.7120798
32. Kamiya, T., Kusumoto, S., Inoue, K.: CCFinder: a multilinguistic token-based code clone detection system for large scale source code. IEEE Trans. Softw. Eng. **28**(7), 654–670 (2002). https://doi.org/10.1109/tse.2002.1019480
33. Komondoor, R., Horwitz, S.: Using slicing to identify duplication in source code. In: Cousot, P. (ed.) SAS 2001. LNCS, vol. 2126, pp. 40–56. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-47764-0_3
34. Li, M., et al.: LIBD: scalable and precise third-party library detection in android markets. In: 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE), pp. 335–346. IEEE (2017)
35. Luo, L., Ming, J., Wu, D., Liu, P., Zhu, S.: Semantics-based obfuscation-resilient binary code similarity comparison with applications to software and algorithm plagiarism detection. IEEE Trans. Softw. Eng. **43**(12), 1157–1177 (2017). https://doi.org/10.1109/tse.2017.2655046
36. Moghaddam, H.M., et al.: Watching you watch. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security - CCS 2019. ACM Press (2019). https://doi.org/10.1145/3319535.3354198, https://doi.org/10.1145/3319535.3354198
37. MSSUN: Android Apps Crawler (2014). https://github.com/mssun/android-apps-crawler
38. MTS, F.: Global monitoring platform (2019). https://www.friendmts.com/channel-protection/global-monitoring-platform/
39. Nikas, A., Alepis, E., Patsakis, C.: I know what you streamed last night: On the security and privacy of streaming. Digit. Invest. **25**, 78–89 (2018). https://doi.org/10.1016/j.diin.2018.03.004, https://doi.org/10.1016/j.diin.2018.03.004
40. UK Intellectual Property Office: Guidance about illicit streaming devices (2017). https://www.gov.uk/government/publications/illicit-streaming-devices/illicit-streaming-devices
41. OpSec: Antipiracy live streaming protection (2020). https://www.opsecsecurity.com/opsec-online/antipiracy-live-streaming-protection
42. Pandey, P., Aliapoulios, M., McCoy, D.: Iniquitous cord-cutting: An analysis of infringing IPTV services. In: 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). IEEE (2019). https://doi.org/10.1109/eurospw.2019.00054
43. Rafique, M.Z., van Goethem, T., Joosen, W., Huygens, C., Nikiforakis, N.: It's free for a reason: exploring the ecosystem of free live streaming services. In: NDSS (2016)

44. Sandvine: 2017 Internet Phenomena - Subscription Television Piracy (2017). https://www.sandvine.com/hubfs/downloads/archive/2017-global-internet-phenomena-spotlight-subscription-television-piracy.pdf
45. Sounthiraraj, D., Sahs, J., Greenwood, G., Lin, Z., Khan, L.: SMV-hunter: large scale, automated detection of SSL/TLS man-in-the-middle vulnerabilities in android apps. In: In Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS 2014). Citeseer (2014)
46. Wikipedia: Jaccard index (2020). https://en.wikipedia.org/wiki/Jaccard_index
47. Wikipedia: Kotlin (programming language) (2020). https://en.wikipedia.org/wiki/Kotlin_(programming_language)
48. Wikipedia: Md5 (2020). https://en.wikipedia.org/wiki/MD5
49. Wikipedia: pcap (2020). https://en.wikipedia.org/wiki/Pcap
50. Zhang, Y., Luo, X., Yin, H.: DexHunter: toward extracting hidden code from packed android applications. In: Computer Security - ESORICS 2015, pp. 293–311. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24177-7_15