Identifying HTTPS-Protected Netflix Videos in Real-Time

Andrew Reed, Michael Kranch
Dept. of Electrical Engineering and Computer Science
United States Military Academy at West Point
West Point, New York, USA
{andrew.reed, michael.kranch}@usma.edu

ABSTRACT

After more than a year of research and development, Netflix recently upgraded their infrastructure to provide HTTPS encryption of video streams in order to protect the privacy of their viewers. Despite this upgrade, we demonstrate that it is possible to accurately identify Netflix videos from passive traffic capture in real-time with very limited hardware requirements. Specifically, we developed a system that can report the Netflix video being delivered by a TCP connection using only the information provided by TCP/IP headers.

To support our analysis, we created a fingerprint database comprised of 42,027 Netflix videos. Given this collection of fingerprints, we show that our system can differentiate between videos with greater than 99.99% accuracy. Moreover, when tested against 200 random 20-minute video streams, our system identified 99.5% of the videos with the majority of the identifications occurring less than two and a half minutes into the video stream.

Keywords

privacy; traffic analysis; dynamic adaptive streaming over HTTP; Netflix

1. INTRODUCTION

As the leading provider of streaming video content in a growing industry, Netflix accounts for more than a third of all traffic in North America [11]. In an effort to improve privacy for their viewers, Netflix recently upgraded their Open Connect infrastructure to provide HTTPS encryption of video content in addition to their ongoing use of HTTPS to protect login and billing information [9]. This new use of HTTPS prevents eavesdroppers from conducting deep packet inspection (DPI), i.e. inspecting HTTP headers and payload data, in order to determine the video that is being streamed. While an improvement, there is a previously disclosed traffic analysis attack that does not rely on DPI to identify the traffic's video content [10].

Since the addition of HTTPS adds a negligible amount of overhead to each video segment, we demonstrate that the aforementioned traffic analysis attack also works against HTTPS-protected Netflix videos. We then improve upon the previous work by fully automating the fingerprint creation process, thereby enabling us to create an extensive collection of Netflix fingerprints which we then use to conduct a robust assessment of the attack. Finally, we developed a network appliance that can, in real-time, identify HTTPS-protected Netflix videos using IP and TCP headers obtained from passive capture of network traffic.

Our primary contributions are:

- A dataset that contains the fingerprints for 42,027 Netflix videos.
- An automated crawler that creates Netflix video fingerprints.
- A method to identify Netflix videos in real-time that does not rely on application-layer information.

We have made our code available at [4]. The rest of our paper is organized as follows. In Section 2, we describe the previous work that we leverage in our paper. In Section 3, we detail our method for obtaining Netflix fingerprints, and we explain our video identification pipeline in Section 4. Section 5 describes our testing and results. Related work is reviewed in Section 6 and suggestions for future work are outlined in Section 7.

2. BACKGROUND

Our paper builds upon the work conducted by Reed and Klimkowski in [10] concerning Netflix's vulnerability to traffic analysis. We also leverage two tools, adudump [14] and OpenWPM [3], in order to enhance our methods for traffic capture and fingerprint creation. We describe these in detail below.

2.1 DASH and VBR Information Leakage

For browser-based streaming, Netflix first encodes their videos as variable bitrate (VBR) MPEG4 and then streams them using Dynamic Adaptive Streaming over HTTP (DASH) via Microsoft Silverlight [6]. In [10], Reed and Klimkowski show that this combination of DASH and VBR can produce sequences of video segment sizes (i.e. fingerprints) that are unique for each video. They also show that this uniqueness is especially true for Netflix, as Netflix allows for a higher degree of bitrate variation when encoding content compared to other streaming services.

Furthermore, [10] demonstrates that these fingerprints can be created for each encoding of a video by parsing the metadata contained at the beginning of each MPEG4 video file. Within each encoding's metadata is a data structure, referred to as the segment index box (sidx), which lists the sizes for each video segment in the file [5]. Since Silverlight needs this information to generate its HTTPS GET requests, the metadata portion of each MPEG4 is requested at the start of the stream. Thus, a researcher

---

The views expressed herein are those of the authors and do not reflect the position of the United States Military Academy, the Department of the Army, or the Department of Defense.

CODASPY'17, March 22-24, 2017, Scottsdale, AZ, USA
ACM 978-1-4503-4523-1/17/03
DOI: http://dx.doi.org/10.1145/3029806.3029821

can build the fingerprints for a video by capturing the metadata transmitted during the first few seconds of a stream; it is not necessary to watch the entire video.

Reed and Klimkowski then outline an identification algorithm for Netflix videos from wireless traffic captures. This identification is done using a six-dimensional kd-tree [1] that stores every two-minute sliding window of each Netflix fingerprint. The six dimensions summarize the overall "size" and "shape" of each window in a way that supports range searches when attempting to identify a portion of a wireless capture. Each range search of the kd-tree produces a shortlist of candidate fingerprint windows that are then checked against the wireless capture using Pearson's product-moment correlation coefficient (r) to determine if a candidate matches the capture.

Although our paper is focused on the wired capture of traffic, Netflix's use of HTTPS presents challenges that are similar to those posed by WPA2-encrypted wireless traffic: (i) application layer data cannot be extracted and (ii) HTTP headers and TLS add overhead to each stream. As shown in Figure 1, however, this additional overhead is quite small and has an almost imperceptible effect on the overall fingerprint. Thus, we harness both the kd-tree and the fingerprint creation technique from [10].

2.2 adudump

adudump[1], presented in [14], is a command line program that uses TCP sequence and acknowledgement numbers to infer the sizes of the application data units (ADUs) transferred over each TCP connection. For example, the 171st video segment of *Home* (3830 kbps encoding) is 2,812,073 bytes. Thus, if a web browser were to send a GET request for this particular segment, adudump would report that the Netflix server responded with a 2,817,667-byte ADU. The additional bytes reported by adudump can be attributed to a 519-byte HTTP header and 5,075 bytes of TLS overhead. Had this same HTTPS response been captured by a program such as tcpdump, then the resultant capture would instead contain the 1,930 individual packets sent by the Netflix server to the browser.

When adudump is run on either a live network interface or a passive network tap, it will log the ADUs for each TCP connection in real-time and print them to stdout. Table 1 shows adudump's output when tracing a sample Netflix video stream. Notice that the direction alternates with every ADU and that all of the outbound ADUs, i.e. the HTTP GETs, are 755-756 bytes, whereas the inbound ADUs, i.e. the video segments, are of varying sizes. This behavior is a direct result of Netflix's use of DASH with VBR.

Since adudump is able to discern between the successive video segments of an HTTPS-protected Netflix stream and report their individual sizes in real-time, we leverage it to provide the input to our identification algorithm. Using adudump as input allows us to conduct kd-tree range searches in Section 4.3 that are tighter than those used in [10].

2.3 OpenWPM

Open Web-Privacy-Measurement (OpenWPM) is a framework for conducting large scale, repeatable web measurement studies [3]. At its core, OpenWPM is simply a Firefox browser that is automated using the Selenium automated browser [13]. For example, OpenWPM takes a list of URLs as input and will visit those sites one at a time until complete. OpenWPM also features several hooks for data collection including an in-band man-in-the-middle proxy [7]. When connecting to a site, the Firefox browser forwards all traffic directly through this proxy. This traffic includes HTTPS which is initially encrypted using the proxy's digital certificate. The proxy then makes an outbound connection to the destination server on behalf of the initial Firefox request. The end state is that all traffic, even encrypted traffic, can be recorded as unencrypted data by the proxy.

[1] For access to adudump, please contact Dr. Jeff Terrell at info@altometrics.com.

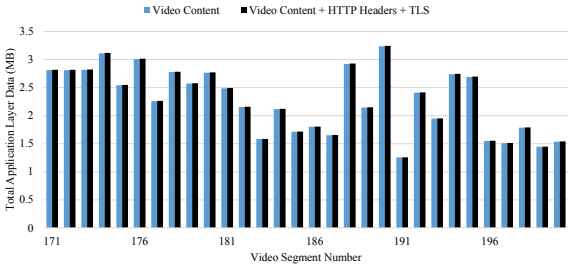

Figure 1: Netflix video overhead due to HTTP headers and TLS (*Home*, 3830 kbps encoding).

Table 1: adudump trace of *Home* (3830 kbps encoding). These are segments 171-180 from Figure 1.

Timestamp	Local PC	Dir.	Netflix Server	Size (B)
1471357732.77583	134.240.17.111.31177	>	198.45.63.167.443	756
1471357736.70148	134.240.17.111.31177	<	198.45.63.167.443	2817667
1471357736.77902	134.240.17.111.31177	>	198.45.63.167.443	756
1471357740.89304	134.240.17.111.31177	<	198.45.63.167.443	2816159
1471357740.97057	134.240.17.111.31177	>	198.45.63.167.443	756
1471357744.45695	134.240.17.111.31177	<	198.45.63.167.443	2822089
1471357744.53453	134.240.17.111.31177	>	198.45.63.167.443	756
1471357748.76052	134.240.17.111.31177	<	198.45.63.167.443	3117490
1471357748.83926	134.240.17.111.31177	>	198.45.63.167.443	756
1471357752.72718	134.240.17.111.31177	<	198.45.63.167.443	2548098
1471357752.80466	134.240.17.111.31177	>	198.45.63.167.443	756
1471357756.87447	134.240.17.111.31177	<	198.45.63.167.443	3014236
1471357756.95195	134.240.17.111.31177	>	198.45.63.167.443	756
1471357760.48768	134.240.17.111.31177	<	198.45.63.167.443	2263764
1471357760.56593	134.240.17.111.31177	>	198.45.63.167.443	756
1471357764.73616	134.240.17.111.31177	<	198.45.63.167.443	2782180
1471357764.81363	134.240.17.111.31177	>	198.45.63.167.443	755
1471357768.73659	134.240.17.111.31177	<	198.45.63.167.443	2577683
1471357768.81421	134.240.17.111.31177	>	198.45.63.167.443	756
1471357772.97218	134.240.17.111.31177	<	198.45.63.167.443	2770492

3. ACQUIRING FINGERPRINTS

In order to identify an unknown video's traffic in real-time, we first need a database of fingerprints of known video traffic to compare against our captured traffic. This database collection requires three steps: (i) identify a unique URL for every Netflix video, (ii) watch each video to generate the set of unique fingerprints, and (iii) organize these fingerprints in a searchable database.

3.1 Initial Crawl to Gather Video URLs

In order to generate these fingerprints, we first mapped every available video on Netflix. We took advantage of Netflix's search feature to do this mapping by conducting iterative search queries to enumerate all of Netflix's videos. This enumeration was done by visiting https://www.netflix.com/search/<value> where <value> was 'a', then 'b', etc. and then parsing the returned HTML into a list of videos with matching URLs. We also searched for videos by category using the list provided at [8].

These categories can be accessed by browsing to https://www.netflix.com/browse/genre/<id> where <id> is the category number (e.g. 6548 for Comedies, 4814 for Miniseries, and 75405 for Zombie Horror Movies).

Note that these initial URLs could be either (i) a direct link to a movie or (ii) a link to a TV show's episode list. Thus, it is necessary to then visit each scraped URL to determine if it is either a movie or an episode list. If it is the latter, then we send a series of JSON requests to Netflix to enumerate every episode's URL by season. In total, we scraped 42,169 unique video URLs using these two techniques in our two months of Netflix crawling.

3.2 Automated Viewing to Acquire Metadata

In order to record the fingerprints for each video, we made several modifications to the provided OpenWPM platform. This platform traditionally takes a list of URLs as input and then automates browsing to these URLs one at a time in succession as soon as the previous site is loaded. The term loaded includes all of the static and dynamic (JavaScript created) HTML objects in the Document Object Model (DOM) standard [2]. In Netflix's case, this loading meant that OpenWPM would download the DOM, consisting primarily of the background image and the Silverlight plugin, but it would not wait for the video to play. As such, we modified OpenWPM to incorporate a delay in order to allow Silverlight time to request the metadata for a video's various encodings before moving on to the next URL.

We also added a module within the proxy to identify the sidx from the requested video's metadata, parse the sidx into fingerprints, and then save the resultant fingerprints into our database. Placing this module within the proxy allowed us to minimize the storage requirement of the database. In general, the proxy would save 7-10 different fingerprints for the various encoding rates within the first 20 seconds of each video stream (acquiring the high definition streams within the first 20 seconds is dependent upon network conditions). Using OpenWPM across four Macbook Pros, we were able to crawl all forty-two thousand URLs within four days.

3.3 Video Database Statistics

Table 2 lists various statistics for the fingerprints in our collection. In total, we collected 330,364 fingerprints from 42,027 videos (average of 7.86 fingerprints per video) which we store in a 1.37 GB text file. As shown, 92% of Netflix's catalog is comprised of television shows. Additionally, Figure 2 shows the breakdown of our database by bitrate. As mentioned in [10], Netflix has historically encoded their browser-based videos at 235, 375, 560, 750, 1050, 1750, 2350, and 3000 kbps, which correspond to the peaks in Figure 2. The emergence of bitrates that fall outside these historic ranges can likely be attributed to the fact that Netflix has recently made improvements in their encoding efficiency [11].

4. VIDEO IDENTIFICATION PIPELINE

4.1 Capturing and Filtering Traffic

Our pipeline begins with adudump, which is run on either a live network interface, a passive network tap (e.g. Endace DAG card), or a pcap file. As adudump logs ADUs, we use a filter to ignore all ADUs except those that are sent from a server on port 443. Additionally, we ignore ADUs that are smaller than 200,000 bytes, as these *could* be audio segments (audio segments are sometimes requested via the same TCP connection used for video segments).

Table 2: Database statistics.

Total Videos			Average Length (h:mm:ss)		
All	Movies	Shows	All	Movies	Shows
42,027	3,247	38,780	0:38:54	1:33:30	0:34:17

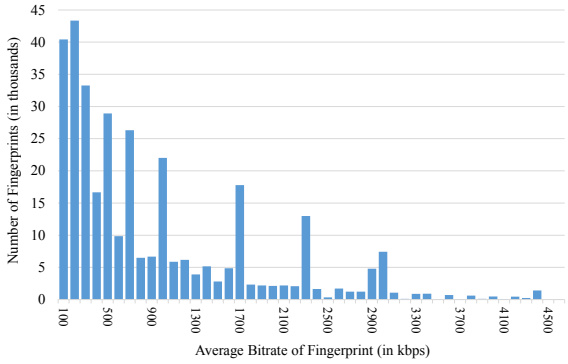

Figure 2: Number of fingerprints by average bitrate. The results are shown in 100 kbps bins. There are 146 fingerprints that exceed 4600 kbps that are not depicted.

4.2 Tracking TCP Connections

We track the following data for each HTTPS TCP connection:

- Timestamp of the first ADU
- Timestamp of the most recent ADU
- A deque containing the sizes of the 30 most recently logged ADUs
- A Boolean to track whether the HTTPS TCP connection has been identified as a Netflix video (initial value: False)
- The title of the video being streamed (initial value: blank)

Once a TCP connection's deque reaches 30 ADUs, and upon the receipt of each new ADU thereafter, we send the deque to the kd-tree for identification.

4.3 kd-Tree Search

Similar to [10], we create a 6D key for each 30-ADU window and conduct a range search of the kd-tree to retrieve a shortlist of potential matches. The ranges for each search are as follows:

- **1st Dimension Min** $= \frac{Total\ Received}{1.0019} - (30 * 525\ bytes)$
- **1st Dimension Max** $= \frac{Total\ Received}{1.0017} - (30 * 515\ bytes)$
- **2nd through 6th Dimension Min**: -0.0001
- **2nd through 6th Dimension Max**: +0.0001

Our 1st dimension ranges are based on these two observations of Netflix traffic:

- HTTP headers add ~520 bytes to each video segment.
- TLS overhead adds ~0.18% to the combined video content plus HTTP headers.

Our ranges for the 2nd through 6th dimensions are based on our observation that the additional overhead from HTTP headers and TLS has a minimal effect on the distribution of data throughout a given 30-ADU window. For instance, the data distributions for the two series shown in Figure 1 differ by no more than ±0.000013.

Once the range search returns a shortlist of candidates, we iterate through the shortlist and calculate Pearson's product-moment

correlation coefficient (r) between each candidate and the 30-ADU window. If a candidate results in an $r \geq 0.9999$, then we report the candidate as a match and cease iterating through the shortlist.

Overall, our search of the kd-tree incorporates two changes to Reed and Klimkowski's algorithm. First, since adudump is able to delineate the data belonging to adjacent video segments, it is no longer necessary to identify and discard outliers, as done in the original algorithm's Stage 2. Second, we eliminated Stage 3 as it is now possible to identify a given Netflix stream from a single matching window (see Section 5.2).

4.4 Reporting Videos

If a match is found for a given 30-ADU window, then the *entire* HTTPS TCP connection is marked as "identified" and the video title is saved. It is possible to classify an entire TCP connection in this manner since it will not be reused for consecutive videos (e.g. if a Netflix account is configured to play the next episode automatically). Thus, it is no longer necessary to conduct queries for a TCP connection once it has been identified. That being said, one can determine if and when the Netflix player switches between quality levels by continuing to send updated deques to the kd-tree and logging the bitrate of each matched window. In Section 5.3.3, we refer to these as the *efficient mode* and the *exhaustive mode*, respectively.

Once a TCP connection has been inactive for a period of time (we use a 2-minute period), we delete its data from the list of tracked connections. As the connection is being deleted, we check to see if it has been flagged as a Netflix video. If it has, then we report the TCP connection's 4-tuple, the timestamps of the first and last ADUs, and the video title.

5. EVALUATION

We implemented our algorithm using a client-server architecture.

- **Server**: The server, which we implemented in Java, is tasked with performing the identification steps outlined in Section 4.3.
- **Client**: The client runs as a Linux command pipeline consisting of adudump and two Python scripts: (i) a script to filter the output from adudump and (ii) a script to perform the tasks outlined in Sections 4.2 and 4.4.

5.1 Server Specifications

Table 3 lists the hardware and software that we used for the server. Given our configuration, the server requires 15 minutes to load the kd-tree with all of the Netflix video fingerprints. Once loaded, the database contains 184,248,110 individual windows.

5.2 Assessing Video Uniqueness

In order to assess the uniqueness of the Netflix fingerprints, we conducted the same search described in Section 4.3 for each window in the database and tallied the number of matches returned for each. As shown in Figure 3, 184,246,173 windows return a single result, i.e. more than 99.9989% of all windows are unique.

Interestingly, 126 windows do not even return themselves. Upon further inspection, we found that these windows stem from two movies, *2001: A Space Odyssey* and *The Gospel Road: A Story of Jesus*, both of which have lengthy periods where the screen is completely dark, thereby resulting in "flat" windows that consist of 30 identically-sized segments. Since Pearson's r cannot be computed for a single point, these flat windows cannot be correlated.

Table 3: Identification Server Specifications.

Operating System	Linux Mint 17.3 MATE
Processors	2x Quad-Core Intel Xeon 2.0 GHz
Memory	32 GB
Manufacture Date	November 2007
Java Version	1.8
Java Heap Allocation	30 GB

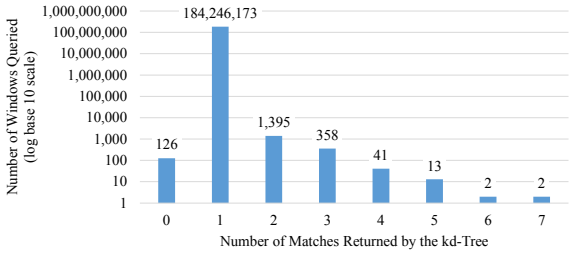

Figure 3: Number of matches returned by the kd-Tree when conducting a search for each window.

Thus, there remains 1,811 windows that return erroneous results, ranging from 1 to 6 mistaken matches. More specifically, these 1,811 windows produce 1,154 unique *pairings* where two windows cannot be distinguished by our algorithm. Of these pairings, 1,053 consist of two windows that are actually the same timeframe of the same video, but from different bitrates. From the remaining 101 pairings (Appendix A), 73 stem from identical footage that exists in multiple videos (e.g. the end credits of *Masha and the Bear* episodes). Only 28 pairings, comprised of 25 distinct windows, represent confusion between unrelated footage. These windows are problematic because, although they primarily consist of complete darkness, they contain enough variation to support the calculation of Pearson's r (i.e. they do not result in a single point, as do the "flat" windows mentioned previously).

In sum, of 184,248,110 total windows:

- 184,246,173 (99.9989%) windows are unique
- 1,786 (0.00097%) windows have the potential to match either (i) the same window from another bitrate or (ii) the same footage in a related video
- 126 (0.000068%) windows are unsearchable due to complete darkness
- 25 (0.000014%) windows have the potential to match an unrelated window due to mostly dark scenes

5.3 Full System Assessment

5.3.1 Experimental Design

For the full system assessment, we first produced a list of 100 randomly-selected Netflix videos that was then given to OpenWPM/Firefox running on two Macbook Pros. Both Macbooks streamed the entire list, with each video being "watched" for 20 minutes before switching to the next video. We chose this duration based on our finding from Table 2 that the vast majority of Netflix content consists of television shows. Considering that (i) television shows that air in 30-minute time slots typically contain 22-24 minutes of video, and that (ii) the Netflix player will skip a show's introduction if an account is set

to "auto-play" the next episode, a duration of 20 minutes per video approximates the amount of traffic generated by a viewer watching a half-hour show.

As the videos were being streamed, we used adudump to capture all network traffic traversing the Internet-facing link shared by the two Macbooks. Once complete, we then used our program to process the adudump capture. As mentioned in Section 4.4, our program displays the (i) video title, (ii) start time, and (iii) end time for each *TCP connection* that produces at least one window that matches a fingerprint. For testing purposes, we also configured our program to display the time at which the first window was identified for a given TCP connection[2].

5.3.2 Results

Our full results are listed in Appendix B. When referencing individual video results, the letter indicates which laptop (A or B) generated the stream and the number indicates which video (1-100) was "watched." To summarize our results:

- Our program identified 199 of 200 video streams (99.5%).
- Video B57 generated two, back-to-back TCP connections. Since these connections were separated by only eight seconds, we combined them as a single report. The remaining 198 identifications consisted of a single TCP connection.
- As shown in Figure 4, 51% of the videos were identified before 2:30 had elapsed.
- On average, initial identifications occurred at 3:55, with the earliest occurring at 2:00 and the latest occurring at 12:04.
- Excluding video B85, our program identified an average of 19:19 of each video, with a minimum of 14:39 and a maximum of 20:00 (i.e. the full video).
- Our program identified the full video in 124 instances (62%).
- Across a total of 240,000 seconds' worth of "viewing activity," our program identified 230,698 seconds (96.12%).

Tests such as B63 highlight the effectiveness of our technique. In B63, our program was not able to identify the video until 12:04 into the stream. Despite B63's late identification, our program was still able to identify the full 20-minute timeframe since the underlying TCP connection lasted for the entire duration of the stream.

5.3.3 System Capacity

In addition to assessing our program's accuracy, we also tested our system to estimate the maximum number of concurrent Netflix streams that it can support in real-time. This assessment was done for both modes of operation: *exhaustive* and *efficient*. As mentioned in Section 4.4, in *exhaustive mode* our program will continue to send windows to the kd-tree even after a TCP connection has been identified, as opposed to *efficient mode* which does not send windows for identified TCP connections.

[2] In order for our program to successfully identify a video, the Netflix stream must stabilize at a quality level for 30 consecutive video segments. If bandwidth conditions are unstable at the beginning of a stream, then our program will not be able to make an early identification.

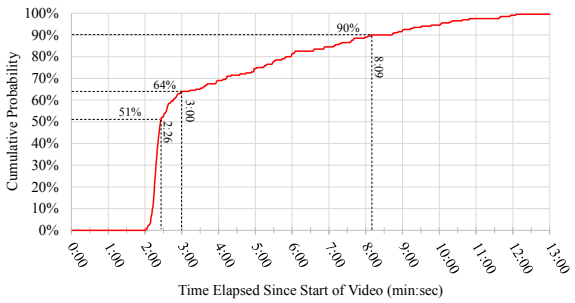

Figure 4: Cumulative probability of identifying a video before a specified amount of time has elapsed.

To begin, we ran our program's *exhaustive mode* 500 times using the adudump capture from Section 5.3.1 and averaged its execution time, yielding an average of 6.56 seconds per iteration. Given this capture, *exhaustive mode* conducted 57,679 window searches per iteration. Thus, our system identified 8,792 windows per second. Furthermore, since a Netflix stream will only generate a new window search every four seconds (i.e. once per video segment received), we estimate that *exhaustive mode* is capable of supporting approximately 35,000 concurrent Netflix streams on our system.

We then ran the same test for *efficient mode*, arriving at an average of 2.48 seconds per iteration, with 12,121 window searches per iteration, i.e. 2.65 times faster than *exhaustive mode* with 21% the number of window searches being conducted. This reduction in the number of windows searches stems from the fact that the Netflix videos in our capture are first identified at 3:55 into the video (on average), which is 19.6% into the 20 minute stream. Overall, we estimate that *efficient mode* is capable of supporting over 92,000 concurrent Netflix streams under similar network conditions. In scenarios where the available bandwidth is constantly fluctuating, Netflix streams might have difficulty finding a stable quality level, thereby reducing *efficient mode's* ability to identify a video early in the stream. Conversely, well-provisioned networks might allow for earlier identifications, thereby increasing *efficient mode's* performance.

6. RELATED WORK

In [16], Zhang et al. provide an overview of the state-of-the-art in traffic classification and state that modern approaches generally rely on (i) machine learning to (ii) identify the applications that are present on a network. Our work differs from these techniques as we neither employ machine learning nor attempt to identify a broad set of network applications. Instead, we focus on identifying the *content* delivered by a *single service*.

Thus, our work is most similar to techniques that exploit the potential for VBR encoding to leak the contents of a particular flow, despite the use of encryption. For instance, earlier work in our domain by Saponas et al. [12] demonstrated that Slingbox video streams could be identified using throughput analysis and a Discrete Fourier Transform-based attack. In another display of VBR data leakage, White et al. [15] demonstrated an attack that reconstructs the transcript of an encrypted VoIP call from its packet sizes.

7. CONCLUSION AND FUTURE WORK

In this paper, we have shown that Netflix's recent upgrade to HTTPS does little to protect the privacy of their users against a passive traffic analysis attack, as we were able to identify 99.5%

of the videos in our tests. Additionally, we presented an automated method to amass Netflix video fingerprints and we conducted a thorough analysis of our database to assess the uniqueness of each fingerprint. We were able to implement our identification pipeline as a program that runs in real-time on limited hardware.

Our work reiterates warnings from previous researchers that VBR encoding can leak details about the underlying content, thereby undermining efforts to protect privacy with encryption. This vulnerability is magnified for DASH streams, as the sequential HTTP GETs generated by DASH correspond to video segment data that can be found in an MPEG4's metadata. As streaming video continues to grow, we believe that streaming services and network researchers should work to solve the privacy issues inherent to DASH and VBR encoding.

To that end, we believe that Netflix could defend against passive traffic analysis by ensuring that the byte-range portion of the HTTP GETs sent by the browser do not perfectly align with individual video segment boundaries. For instance, the browser could average the size of several consecutive segments and send HTTP GETs for this average size. As an alternative approach, the browser could randomly combine consecutive segments and send HTTP GETs for the combined video data. Designing obfuscation techniques for VBR DASH streams that do not degrade video quality remains a potential area for future research.

8. REFERENCES

[1] J. L. Bentley. Multidimensional Binary Search Trees Used for Associative Searching. In *Communications of the ACM*, September 1975.

[2] DOM Standard, https://dom.spec.whatwg.org/.

[3] S. Englehardt and A. Narayanan. Online Tracking: A 1-Million-Site Measurement and Analysis. In *ACM Conference on Computer and Communications Security*, 2016.

[4] GitHub Repository, https://github.com/andrewreed.

[5] ISO/IEC 14496-12:2012, http://standards.iso.org/ittf/PubliclyAvailableStandards/c061988_ISO_IEC_14496-12_2012.zip.

[6] Microsoft Silverlight, https://www.microsoft.com/silverlight.

[7] mitmproxy, https://mitmproxy.org.

[8] Netflix has tons of hidden categories — here's how to see them, http://mashable.com/2016/01/11/netflix-search-codes.

[9] The Netflix Tech Blog: Protecting Netflix Viewing Privacy at Scale, http://techblog.netflix.com/2016/08/protecting-netflix-viewing-privacy-at.html.

[10] A. Reed and B. Klimkowski. Leaky Streams: Identifying Variable Bitrate DASH Videos Streamed over Encrypted 802.11n Connections. In *IEEE Consumer Communications and Networking Conference*, 2016.

[11] Sandvine Report: Netflix's Encoding Optimizations Result In North American Traffic Share Decline, https://www.sandvine.com/pr/2016/6/22/sandvine-report-netflix-encoding-optimizations-result-in-north-american-traffic-share-decline.html.

[12] T. S. Saponas, J. Lester, C. Hartung, S. Agarwal, and T. Kohno. Devices that Tell on You: Privacy Trends in Consumer Ubiquitous Computing. In *USENIX Security Symposium*, 2007.

[13] Selenium, http://www.seleniumhq.org.

[14] J. Terrell, K. Jeffay, F. D. Smith, J. Gogan, and J. Keller. Passive, Streaming Inference of TCP Connection Structure for Network Server Management. In *IEEE International Traffic Monitoring and Analysis Workshop*, 2009.

[15] A. White, A. Matthews, K. Snow, and F. Monrose. Phonotactic Reconstruction of Encrypted VoIP Conversations: Hookt on fon-iks. In *IEEE Symposium on Security and Privacy*, 2011.

[16] J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu. Robust Network Traffic Classification. In *IEEE/ACM Transactions on Networking*, August 2015.

APPENDIX

A. Incorrect Pairings.

Shaded pairings denote identical footage.

Video 1			Video 2			
Title	Average Bitrate (kbps)	Starting Segment	Title	Average Bitrate (kbps)	Starting Segment	Reason for Confusion
2001: A Space Odyssey	119	1	*The Gospel Road: A Story of Jesus*	235	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	119	1	*The Gospel Road: A Story of Jesus*	371	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	119	1316	*The Agony and the Ecstasy*	235	916	Both videos have an intermission consisting of darkness
2001: A Space Odyssey	119	1316	*The Agony and the Ecstasy*	371	916	Both videos have an intermission consisting of darkness
2001: A Space Odyssey	119	2188	*The Agony and the Ecstasy*	235	2053	Both videos end with several minutes of darkness
2001: A Space Odyssey	119	2188	*The Agony and the Ecstasy*	371	2053	Both videos end with several minutes of darkness
2001: A Space Odyssey	119	2189	*The Agony and the Ecstasy*	235	2054	Both videos end with several minutes of darkness
2001: A Space Odyssey	119	2189	*The Agony and the Ecstasy*	371	2054	Both videos end with several minutes of darkness
2001: A Space Odyssey	119	2190	*The Agony and the Ecstasy*	235	2055	Both videos end with several minutes of darkness
2001: A Space Odyssey	119	2190	*The Agony and the Ecstasy*	371	2055	Both videos end with several minutes of darkness
2001: A Space Odyssey	165	1	*The Gospel Road: A Story of Jesus*	235	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	165	1	*The Gospel Road: A Story of Jesus*	371	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	165	1	*The Gospel Road: A Story of Jesus*	551	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	165	1	*The Gospel Road: A Story of Jesus*	736	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	240	1	*The Gospel Road: A Story of Jesus*	371	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	240	1	*The Gospel Road: A Story of Jesus*	551	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	240	1	*The Gospel Road: A Story of Jesus*	736	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	240	1316	*The Agony and the Ecstasy*	551	916	Both videos have an intermission consisting of darkness
2001: A Space Odyssey	240	1316	*The Agony and the Ecstasy*	736	916	Both videos have an intermission consisting of darkness
2001: A Space Odyssey	380	1	*The Gospel Road: A Story of Jesus*	1029	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	380	1	*The Gospel Road: A Story of Jesus*	1710	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	380	1316	*The Agony and the Ecstasy*	1028	916	Both videos have an intermission consisting of darkness
2001: A Space Odyssey	530	1	*The Gospel Road: A Story of Jesus*	1029	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	530	1	*The Gospel Road: A Story of Jesus*	1710	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	530	1316	*The Agony and the Ecstasy*	1028	916	Both videos have an intermission consisting of darkness
2001: A Space Odyssey	856	1	*The Gospel Road: A Story of Jesus*	1029	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	856	1	*The Gospel Road: A Story of Jesus*	1710	1216	Video 1 begins with several minutes of darkness / Video 2 ends with several minutes of darkness
2001: A Space Odyssey	856	1316	*The Agony and the Ecstasy*	1028	916	Both videos have an intermission consisting of darkness
Las mágicas historias de Plim Plim/Season 1 : Episode 1	1726	360	*The Magical Tales of Plim Plim/Season 1 : Episode 1*	1726	360	Both videos contain identical footage (Spanish vs. English dialogue)
Masha and the Bear/Season 1 : Episode 1	1576	312	*Masha and the Bear/Season 1 : Episode 2*	1634	302	Identical end credits
Masha and the Bear/Season 1 : Episode 1	1576	316	*Masha and the Bear/Season 1 : Episode 5*	1335	309	Identical end credits
Masha and the Bear/Season 1 : Episode 1	1576	317	*Masha and the Bear/Season 1 : Episode 5*	1335	310	Identical end credits
Masha and the Bear/Season 1 : Episode 1	1576	318	*Masha and the Bear/Season 1 : Episode 2*	1634	308	Identical end credits
Masha and the Bear/Season 1 : Episode 1	1576	324	*Masha and the Bear/Season 1 : Episode 2*	1634	314	Identical end credits
Masha and the Bear/Season 1 : Episode 1	1576	326	*Masha and the Bear/Season 1 : Episode 5*	1335	319	Identical end credits
Masha and the Bear/Season 1 : Episode 1	2140	317	*Masha and the Bear/Season 1 : Episode 5*	2082	310	Identical end credits
Masha and the Bear/Season 1 : Episode 2	1634	304	*Masha and the Bear/Season 1 : Episode 5*	1335	307	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	302	*Masha and the Bear/Season 1 : Episode 5*	303	305	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	303	*Masha and the Bear/Season 1 : Episode 5*	303	306	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	304	*Masha and the Bear/Season 1 : Episode 5*	303	307	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	305	*Masha and the Bear/Season 1 : Episode 5*	303	308	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	306	*Masha and the Bear/Season 1 : Episode 5*	303	309	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	307	*Masha and the Bear/Season 1 : Episode 5*	303	310	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	308	*Masha and the Bear/Season 1 : Episode 5*	303	311	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	309	*Masha and the Bear/Season 1 : Episode 5*	303	312	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	310	*Masha and the Bear/Season 1 : Episode 5*	303	313	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	311	*Masha and the Bear/Season 1 : Episode 5*	303	314	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	312	*Masha and the Bear/Season 1 : Episode 5*	303	315	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	313	*Masha and the Bear/Season 1 : Episode 5*	303	316	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	314	*Masha and the Bear/Season 1 : Episode 5*	303	317	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	315	*Masha and the Bear/Season 1 : Episode 5*	303	318	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	316	*Masha and the Bear/Season 1 : Episode 5*	303	319	Identical end credits
Masha and the Bear/Season 1 : Episode 2	306	317	*Masha and the Bear/Season 1 : Episode 5*	303	320	Identical end credits
Masha and the Bear/Season 1 : Episode 3	3098	317	*Masha and the Bear/Season 1 : Episode 5*	2854	317	Identical end credits
Masha and the Bear/Season 1 : Episode 3	3098	318	*Masha and the Bear/Season 1 : Episode 5*	2854	318	Identical end credits
Masha and the Bear/Season 1 : Episode 4	357	307	*Masha and the Bear/Season 1 : Episode 6*	357	309	Identical end credits
Masha and the Bear/Season 1 : Episode 4	357	310	*Masha and the Bear/Season 1 : Episode 9*	356	312	Identical end credits
Masha and the Bear/Season 1 : Episode 4	573	314	*Masha and the Bear/Season 1 : Episode 6*	574	316	Identical end credits
Masha and the Bear/Season 1 : Episode 4	573	317	*Masha and the Bear/Season 1 : Episode 6*	574	319	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1319	306	*Masha and the Bear/Season 1 : Episode 9*	1191	306	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1319	307	*Masha and the Bear/Season 1 : Episode 9*	1191	307	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1319	308	*Masha and the Bear/Season 1 : Episode 9*	1191	308	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1319	309	*Masha and the Bear/Season 1 : Episode 9*	1191	309	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1319	312	*Masha and the Bear/Season 1 : Episode 9*	1191	312	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1319	313	*Masha and the Bear/Season 1 : Episode 9*	1191	313	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1319	314	*Masha and the Bear/Season 1 : Episode 9*	1191	314	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1319	315	*Masha and the Bear/Season 1 : Episode 9*	1191	315	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1865	308	*Masha and the Bear/Season 1 : Episode 7*	1862	307	Identical end credits
Masha and the Bear/Season 1 : Episode 6	1865	309	*Masha and the Bear/Season 1 : Episode 7*	1862	308	Identical end credits
Masha and the Bear/Season 1 : Episode 6	2686	307	*Masha and the Bear/Season 1 : Episode 9*	2660	307	Identical end credits
Masha and the Bear/Season 1 : Episode 6	2686	308	*Masha and the Bear/Season 1 : Episode 9*	2660	308	Identical end credits
Masha and the Bear/Season 1 : Episode 6	2686	309	*Masha and the Bear/Season 1 : Episode 9*	2660	309	Identical end credits
Masha and the Bear/Season 1 : Episode 6	2686	313	*Masha and the Bear/Season 1 : Episode 9*	2660	313	Identical end credits
Masha and the Bear/Season 1 : Episode 6	2686	314	*Masha and the Bear/Season 1 : Episode 9*	2660	314	Identical end credits
Masha and the Bear/Season 1 : Episode 6	2686	315	*Masha and the Bear/Season 1 : Episode 9*	2660	315	Identical end credits
Masha and the Bear/Season 1 : Episode 6	2686	319	*Masha and the Bear/Season 1 : Episode 9*	2660	319	Identical end credits
Masha and the Bear/Season 1 : Episode 6	2686	320	*Masha and the Bear/Season 1 : Episode 9*	2660	320	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	304	*Masha and the Bear/Season 1 : Episode 9*	3846	304	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	305	*Masha and the Bear/Season 1 : Episode 9*	3846	305	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	306	*Masha and the Bear/Season 1 : Episode 9*	3846	306	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	307	*Masha and the Bear/Season 1 : Episode 9*	3846	307	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	308	*Masha and the Bear/Season 1 : Episode 9*	3846	308	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	309	*Masha and the Bear/Season 1 : Episode 9*	3846	309	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	310	*Masha and the Bear/Season 1 : Episode 9*	3846	310	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	311	*Masha and the Bear/Season 1 : Episode 9*	3846	311	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	312	*Masha and the Bear/Season 1 : Episode 9*	3846	312	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	313	*Masha and the Bear/Season 1 : Episode 9*	3846	313	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	314	*Masha and the Bear/Season 1 : Episode 9*	3846	314	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	315	*Masha and the Bear/Season 1 : Episode 9*	3846	315	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	316	*Masha and the Bear/Season 1 : Episode 9*	3846	316	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	317	*Masha and the Bear/Season 1 : Episode 9*	3846	317	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	318	*Masha and the Bear/Season 1 : Episode 9*	3846	318	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	319	*Masha and the Bear/Season 1 : Episode 9*	3846	319	Identical end credits
Masha and the Bear/Season 1 : Episode 6	3886	320	*Masha and the Bear/Season 1 : Episode 9*	3846	320	Identical end credits
Masha and the Bear/Season 1 : Episode 7	1197	302	*Masha and the Bear/Season 1 : Episode 8*	1093	304	Identical end credits
The Gospel of John: King James Version	1052	670	*The Gospel of John: Reina-Valera 1960*	1052	670	Both videos contain identical footage (English vs. Spanish dialogue)
The Gospel of John: King James Version	1749	1796	*The Gospel of John: Reina-Valera 1960*	1749	1796	Both videos contain identical footage (English vs. Spanish dialogue)
The Gospel of John: King James Version	1749	1970	*The Gospel of John: Reina-Valera 1960*	1749	1970	Both videos contain identical footage (English vs. Spanish dialogue)
The Gospel of John: King James Version	2348	797	*The Gospel of John: Reina-Valera 1960*	2348	797	Both videos contain identical footage (English vs. Spanish dialogue)
The Gospel of John: King James Version	380	1353	*The Gospel of John: Reina-Valera 1960*	380	1353	Both videos contain identical footage (English vs. Spanish dialogue)
The Gospel of John: King James Version	564	1927	*The Gospel of John: Reina-Valera 1960*	564	1927	Both videos contain identical footage (English vs. Spanish dialogue)

B. Full System Assessment Results.

#	Video	Laptop A		Laptop B	
		1st Detection	Total Detected	1st Detection	Total Detected
1	*A Different World/Season 5 : Episode 5*	2:22	18:19	2:11	18:24
2	*American Dad!/Season 9 : Episode 9*	6:35	18:28	2:13	18:26
3	*An Idiot Abroad/Season 1 : Episode 1*	2:21	20:00	2:13	20:00
4	*Angel/Season 2 : Episode 21*	2:18	20:00	2:13	20:00
5	*Anthony Bourdain: Parts Unknown/Season 2 : Episode 4*	2:51	20:00	2:33	20:00
6	*Archer/Season 3 : Episode 8*	2:37	17:30	4:34	17:19
7	*Army Wives/Season 7 : Episode 12*	2:23	20:00	2:15	20:00
8	*Arrested Development/Season 1 : Episode 16*	2:35	18:07	4:11	19:08
9	*Arrow/Season 2 : Episode 10*	2:57	20:00	2:11	20:00
10	*Beauty & the Beast/Season 1 : Episode 21*	3:39	20:00	2:11	20:00
11	*Black Butler/Season 1 : Episode 25*	2:19	20:00	2:11	20:00
12	*Bones/Season 5 : Episode 17*	2:45	20:00	2:14	20:00
13	*Broadchurch/Season 2 : Episode 3*	2:19	20:00	2:15	20:00
14	*Buffy the Vampire Slayer/Season 4 : Episode 3*	2:17	20:00	2:14	20:00
15	*Californication/Season 2 : Episode 2*	5:28	20:00	2:23	20:00
16	*Casper's Scare School/Season 1 : Episode 9*	2:17	20:00	2:21	19:13
17	*Cheers/Season 7 : Episode 9*	5:13	20:00	7:10	20:00
18	*Chuck/Season 2 : Episode 7*	2:46	19:58	2:14	20:00
19	*Chuggington/Season 3 : Episode 1*	2:17	15:54	2:05	15:52
20	*Courage the Cowardly Dog/Season 4 : Episode 11*	2:13	18:38	2:09	18:45
21	*Criminal Minds/Season 3 : Episode 8*	2:31	20:00	9:30	20:00
22	*Criminal Minds/Season 5 : Episode 3*	2:16	20:00	4:20	20:00
23	*Cupcake Wars Collection/Collection 2 : Episode 14*	2:30	20:00	2:29	20:00
24	*Digimon: Digital Monsters/Season 2 : Episode 20*	2:53	17:54	2:58	17:53
25	*Diners, Drive-Ins and Dives Collection/Collection 2 : Episode 5*	2:17	17:08	3:34	17:12
26	*Dinotopia: The Mini-Series/Season 1 : Episode 3*	2:21	20:00	2:09	20:00
27	*Doc Martin/Season 1 : Episode 1*	2:52	20:00	2:19	20:00
28	*Drop Dead Diva/Season 5 : Episode 7*	2:16	20:00	8:08	20:00
29	*Ella the Elephant/Season 1 : Episode 9*	3:12	19:37	2:20	19:48
30	*Food Network Star/Season 10 : Episode 7*	2:35	20:00	9:00	20:00
31	*Fringe/Season 3 : Episode 19*	6:03	20:00	5:32	20:00
32	*From Dusk Till Dawn: The Series/Season 2 : Episode 8*	2:15	20:00	7:40	20:00
33	*Good Luck Charlie/Season 1 : Episode 1*	2:21	18:54	2:10	19:03
34	*Goosebumps/Season 2 : Episode 7*	2:21	18:06	2:34	18:07
35	*Grounded for Life/Season 2 : Episode 13*	2:41	18:10	6:04	14:39
36	*Hawaii Five-0/Season 1 : Episode 11*	3:22	20:00	2:18	20:00
37	*Hell on Wheels/Season 4 : Episode 3*	2:19	20:00	5:15	20:00
38	*Heroes/Season 4 : Episode 2*	4:53	20:00	4:57	20:00
39	*House of Cards/Season 2 : Episode 8*	10:48	20:00	2:18	20:00
40	*House, M.D./Season 1 : Episode 8*	2:14	20:00	6:51	20:00
41	*House, M.D./Season 3 : Episode 5*	2:15	20:00	10:20	20:00
42	*How I Met Your Mother/Season 6 : Episode 18*	4:13	18:10	2:08	18:12
43	*Johnny Test/Season 6 : Episode 18*	2:17	17:58	2:17	17:59
44	*LEGO: Legends of Chima/Season 1 : Episode 14*	3:57	19:01	7:07	17:58
45	*La Familia P. Luche/Season 1 : Episode 37*	2:12	19:12	2:00	19:16
46	*Lie to Me/Season 2 : Episode 1*	4:06	20:00	2:20	20:00
47	*Liv and Maddie/Season 2 : Episode 21*	2:23	19:36	2:23	19:44
48	*Lo que la vida me robó/Season 1 : Episode 73*	2:25	20:00	2:22	20:00
49	*Madam Secretary/Season 1 : Episode 10*	2:11	20:00	2:17	20:00
50	*Malcolm in the Middle/Season 5 : Episode 18*	2:25	17:57	10:16	18:04
51	*Malcolm in the Middle/Season 6 : Episode 4*	2:15	18:20	2:15	19:27
52	*Marvel's Agents of S.H.I.E.L.D./Season 1 : Episode 20*	3:36	20:00	10:01	20:00
53	*Masha's Tales/Season 1 : Episode 1*	5:27	18:35	2:42	18:33
54	*Mighty Morphin Power Rangers/Season 3 : Episode 22*	2:20	16:05	2:48	16:08
55	*Miss XV/Season 1 : Episode 5*	11:39	20:00	9:46	20:00
56	*My Name Is Earl/Season 1 : Episode 22*	3:30	17:17	2:26	18:18
57	*My Name Is Earl/Season 2 : Episode 8*	7:39	20:00	9:17	20:00
58	*NCIS/Season 1 : Episode 2*	2:36	20:00	2:22	20:00
59	*NCIS/Season 4 : Episode 12*	3:41	20:00	8:59	20:00
60	*New Girl/Season 4 : Episode 19*	5:47	17:59	2:14	18:03
61	*Nurse Jackie/Season 1 : Episode 5*	2:24	20:00	11:55	20:00
62	*One Tree Hill/Season 3 : Episode 9*	5:43	20:00	9:13	20:00
63	*One Tree Hill/Season 4 : Episode 11*	5:29	20:00	12:04	20:00
64	*Power Rangers Jungle Fury/Season 1 : Episode 26*	2:20	18:40	5:58	18:37
65	*Power Rangers in Space/Season 1 : Episode 5*	2:18	16:18	4:44	16:16
66	*Prison Break/Season 1 : Episode 15*	2:20	20:00	10:37	20:00
67	*Rebelde/Season 1 : Episode 10*	3:56	20:00	2:15	20:00
68	*Rebelde/Season 1 : Episode 312*	4:13	20:00	2:18	20:00
69	*Rebelde/Season 1 : Episode 380*	2:16	20:00	2:15	20:00
70	*Rosario + Vampire/Season 2 : Episode 12*	2:19	20:00	2:15	20:00
71	*Rosario Tijeras/Season 1 : Episode 22*	2:20	20:00	2:24	20:00
72	*Scrubs/Season 2 : Episode 19*	2:03	16:55	2:03	17:59
73	*Some Assembly Required/Season 1 : Episode 23*	8:52	19:08	11:37	19:12
74	*Some Assembly Required/Season 1 : Episode 7*	4:57	19:09	2:25	19:14
75	*Some Assembly Required/Season 1 : Episode 9*	2:36	19:09	8:45	19:18
76	*Sons of Anarchy/Season 2 : Episode 2*	2:14	20:00	5:49	20:00
77	*Star Trek/Season 1 : Episode 24*	2:13	20:00	7:58	20:00
78	*Star Trek: Enterprise/Season 4 : Episode 10*	2:38	20:00	7:16	20:00
79	*Stuck in Love*	2:16	20:00	2:15	20:00
80	*Supernatural/Season 6 : Episode 6*	2:09	20:00	2:50	20:00
81	*Sword Art Online II/Season 1 : Episode 6*	6:52	20:00	2:26	20:00
82	*Teresa/Season 1 : Episode 42*	7:20	20:00	4:57	20:00
83	*That '70s Show/Season 1 : Episode 8*	2:24	19:24	2:23	18:27
84	*The Blacklist/Season 1 : Episode 6*	2:19	20:00	10:03	20:00
85	*The Boondocks/Season 2 : Episode 3*	8:03	15:03	**not detected**	
86	*The Carrie Diaries/Season 1 : Episode 9*	2:36	20:00	2:19	20:00
87	*The Day My Butt Went Psycho!/Season 1 : Episode 9*	2:09	18:37	2:17	18:28
88	*The IT Crowd/Season 3 : Episode 6*	3:55	19:29	5:59	19:36
89	*The L Word/Season 4 : Episode 10*	6:34	20:00	5:59	20:00
90	*The Vampire Diaries/Season 1 : Episode 5*	2:14	20:00	7:36	20:00
91	*The X-Files/Season 1 : Episode 22*	2:30	20:00	7:35	20:00
92	*The X-Files/Season 3 : Episode 12*	5:18	20:00	8:43	20:00
93	*Total Drama/Season 1 : Episode 27*	2:13	20:00	2:07	20:00
94	*Trailer Park Boys/Season 5 : Episode 2*	2:13	19:14	2:09	19:05
95	*Transformers: Rescue Bots/Season 2 : Episode 20*	2:12	18:08	2:11	18:00
96	*Tree Fu Tom/Season 2 : Episode 11*	2:21	17:09	2:24	17:07
97	*Trotro/Season 1 : Episode 10*	2:17	17:28	2:04	17:24
98	*Victoria/Season 1 : Episode 165*	5:01	20:00	2:09	20:00
99	*White Collar/Season 5 : Episode 4*	2:15	20:00	2:19	20:00
100	*Xena: Warrior Princess/Season 2 : Episode 10*	2:22	20:00	2:17	20:00