Effective Detection of Multimedia Protocol Tunneling using Machine Learning

Diogo Barradas, Nuno Santos, and Luís Rodrigues, ***INESC-ID, Instituto Superior Técnico, Universidade de Lisboa***

https://www.usenix.org/conference/usenixsecurity18/presentation/barradas

This paper is included in the Proceedings of the 27th USENIX Security Symposium.

August 15–17, 2018 • Baltimore, MD, USA

ISBN 978-1-939133-04-5

Open access to the Proceedings of the 27th USENIX Security Symposium is sponsored by USENIX.

Effective Detection of Multimedia Protocol Tunneling using Machine Learning

Diogo Barradas Nuno Santos Luís Rodrigues
INESC-ID, Instituto Superior Técnico, Universidade de Lisboa
{*diogo.barradas, nuno.m.santos, ler*}*@tecnico.ulisboa.pt*

Abstract

Multimedia protocol tunneling enables the creation of covert channels by modulating data into the input of popular multimedia applications such as Skype. To be effective, protocol tunneling must be unobservable, i.e., an adversary should not be able to distinguish the streams that carry a covert channel from those that do not. However, existing multimedia protocol tunneling systems have been evaluated using ad hoc methods, which casts doubts on whether such systems are indeed secure, for instance, for censorship-resistant communication.

In this paper, we conduct an experimental study of the unobservability properties of three state of the art systems: Facet, CovertCast, and DeltaShaper. Our work unveils that previous claims regarding the unobservability of the covert channels produced by those tools were flawed and that existing machine learning techniques, namely those based on decision trees, can uncover the vast majority of those channels while incurring in comparatively lower false positive rates. We also explore the application of semi-supervised and unsupervised machine learning techniques. Our findings suggest that the existence of manually labeled samples is a requirement for the successful detection of covert channels.

1 Introduction

Multimedia protocol tunneling has emerged as a potentially effective technique to create covert channels which are difficult to identify. In a nutshell, this technique consists of encoding covert data into the video (and / or audio) channel of popular encrypted streaming applications such as Skype without requiring any changes to the carrier application. Systems such as Facet [30], CovertCast [34], and DeltaShaper [2] implement this technique, and introduce different approaches for data modulation that aim at raising the difficulty of an adversary to identify covert data transmissions.

An important property that all these systems strive to achieve is *unobservability*. A covert channel is deemed unobservable if an adversary that is able to scan any number of streams is not able to distinguish those that carry a covert channel from those that do not [20, 23]. Thus, an adversary aims at correctly detecting all streams that carry covert channels, among a set of genuine streams, as effectively as possible. In practice, a multimedia protocol tunneling system that provides a high degree of unobservability prevents an adversary from flagging a large fraction of covert flows (i.e., from attaining a high true positive rate) while flagging a low amount of regular traffic (i.e., while attaining a low false positive rate).

In spite of the efforts to build unobservable systems, the methodology currently employed for their evaluation raises concerns. To assess the unobservability of a system such as Facet, experiments are mounted in order to play regular traffic along with covert traffic, collect the resulting traces, and employ similarity-based classifiers (e.g., relying in the χ^2 similarity function) to determine whether covert traffic can be detected with a low number of false positives [30]. However, each system has been evaluated with a different classifier, making results hard to compare. Furthermore, those studies use just one among the many machine learning (ML) techniques available today. Yet, providing a common ground for assessing the unobservability of multimedia protocol tunneling systems is a relevant problem which, nevertheless, has been overlooked in the literature. Considering that such systems emerged from the need to circumvent Internet censorship, flawed systems may pose life-threatening risks to end-users, e.g., journalists that report news in extreme conditions may be prosecuted, imprisoned, or even murdered if covert channels are detected.

To fill this gap, our goal is to systematically assess the unobservability of existing systems against powerful adversaries making use of traffic analysis techniques based on ML. We aim at understanding which ML techniques are better suited for the purpose of detecting covert chan-

nels in multimedia streams and what are the limitations of such techniques. In particular, we seek to explore ML techniques which have yielded successful results when applied in other domains (e.g., Tor hidden services fingerprinting [22]), but have not yet been studied in the context of covert traffic detection.

In this paper, we present the first experimental study of the unobservability of covert channels produced by state-of-the-art multimedia protocol tunneling systems. We test three systems – Facet, CovertCast, and Deltashaper – using the original code provided by their maintainers. For our study, we take a systematic approach by investigating a spectrum of anomaly detection techniques, ranging from supervised, to semi-supervised and unsupervised, where for each category we explore different classifiers, and investigate the trade-offs involved in the ability to flag a large amount of covert channels while minimizing false positives. From our study, we highlight the following three main contributions.

First, our analysis reveals that some state-of-the-art systems are flawed. In particular, CovertCast flows can be detected with few false positives by an adversary, even when resorting to existing similarity-based classifiers. While the remaining systems exhibit different degrees of unobservability according to their parameterization, we show that none of the currently employed similarity-based classifiers can detect such channels without incurring in large numbers of false positives. We also conclude that one of the existing similarity-based classifiers – using χ^2 distance – consistently outperforms all others in the task of detecting covert channels.

Second, we show that ML techniques based on decision trees and some of their variants are extremely effective at detecting covert traffic with reduced false positive rates. For example, an adversary employing XGBoost would be able to flag 90% of all Facet traffic while erroneously flagging only 2% of legitimate connections. Moreover, the performance of such techniques is very high, meaning that the adversary is able to classify traffic in a few seconds, with a relatively low number of samples per training set, and taking a low memory footprint. Additionally, the use of decision tree-based techniques allows us to understand which traffic features are most important for detecting the functioning of particular multimedia protocol tunneling systems. These findings suggest that, apart from their performance, decision tree-based techniques can provide meaningful insight into the inner workings of these systems and we propose that they should be used for assessing the unobservability of multimedia protocol tunneling systems in the future.

Third, we explore alternative ML approaches for the detection of covert channels when the adversary is assumed to be partially or totally deprived of labeled data. Our findings suggest that unsupervised learning techniques provide no advantage for the classification of multimedia protocol tunneling covert channels, while the application of semi-supervised learning techniques yields a significant fraction of false positives. However, we note that the performance of semi-supervised techniques can be significantly improved through the optimization of parameters or by providing algorithms with extra training data. The study of semi-supervised anomaly detection techniques with an ability to self-tune parameters can be a promising future direction of research which would enable adversaries to detect covert traffic while avoiding the burden of generating and manually label data.

We note that we synthesize a limited number of legitimate and covert traffic samples in laboratory settings for creating our datasets. While this is a common approach for generating datasets for the type of unobservability assessment we conduct in this paper, it is possible that adversaries possessing a privileged position in the network can build a more accurate representation of traffic.

The remainder of our paper is organized as follows. Section 2 presents the methodology of our study. Section 3 presents the main findings of our study regarding the comparison of similarity-based classifiers. Section 4 presents the results obtained when assessing unobservability resorting to decision tree-based classifiers. Section 5 presents our first insights on using semi-supervised and unsupervised anomaly detection techniques for the identification of covert traffic. In Section 6, we discuss obtained results and we present the related work in Section 7. Lastly, we conclude our work in Section 8.

2 Methodology

This section introduces the systems we analyzed, our adversary model, and the experimental setup of our study.

2.1 Systems Under Analysis

Below, we describe three state-of-the-art approaches at multimedia protocol tunneling which serve as a basis for our study. We selected these systems because all of them encode data into video streams, and their code is publicly available for open testing. We note that although these systems have been conceived for the purpose of censorship circumvention, in practice, they may be used for other purposes, such as concealing criminal activity.

Facet [30] allows clients to watch arbitrary videos by replacing the audio and video feeds of Skype videocalls. To watch a video, clients contact a Facet server by sending it a message containing the desired video URL. Afterwards, the Facet server downloads the requested video and feeds its content to microphone and camera emulators. Then, the server places a videocall to the client

transmitting the selected video and audio instead. Thus, clients are not required to install any software in order to use the system. For approximating the traffic patterns of regular videocalls, Facet re-samples the audio frequency and overlays the desired video in a fraction of each frame while the remaining frame area is filled up by a video resembling a typical videocall. Decreasing the area occupied by the concealed video translates into increased resistance against traffic analysis.

CovertCast [34] scrapes and modulates the content of web pages into images which are distributed via live-streaming platforms such as YouTube. Multiple clients can consume the data being transmitted in a particular live stream simultaneously. CovertCast modulates web content by encoding it into colored matrix images. A colored matrix is parameterized by a cell size (adjacent pixels with a given color), the number of bits encoded in each cell (represented with a color), and the rate at which a matrix containing new data is loaded. Clients scrape and demodulate the images served through the live stream extracting the desired web content.

DeltaShaper [2] differentiates itself from the previous systems in that it allows for tunneling arbitrary TCP/IP traffic. This is achieved by modulating covert data into images which are transmitted through a bi-directional Skype videocall. DeltaShaper follows a similar data encoding mechanism to that of CovertCast. However, and similarly to Facet, a colored matrix is overlayed in a fraction of the call screen, on top of a typical chat video running in the background. This overlay, named payload frame, can be carefully parameterized to provide different levels of resistance against traffic analysis. On call start, DeltaShaper undergoes a calibration phase for adjusting its encoding parameters according to the current network conditions in order to preserve unobservability.

2.2 Adversary Model

To study the unobservability properties of the aforementioned systems, we emulate a state-level adversary which will attempt to detect the traffic of multimedia protocol tunneling tools while resorting to different anomaly detection techniques. The providers of encrypted multimedia applications which are used as carriers for covert channels are not assumed to collude with the adversary. Thus, the adversary cannot simply demand application providers to decipher and disclose raw multimedia content which could be easily screened for the presence of covert data. The adversary is also assumed to be unable to control the software installed in the computers of end-users. However, domestic ISPs are assumed to cooperate with the adversary, enabling it to monitor, store and inspect all traffic flows crossing its borders.

An adversary faces an inherent trade-off between the ability to correctly detect a large amount of covert channels and to erroneously flag legitimate flows. Flagging legitimate flows as covert channels is something that the adversary wants to avoid in most practical settings. For example, a censor that aims at blocking flows containing covert channels may not be willing to block large fractions of legitimate calls, that are used daily by companies and business, as these calls may be key for the economy of the censor's regime [17]. Also, law-enforcement agencies may not be willing to risk to falsely flag legitimate actions of citizens as criminal activity.

2.3 Performance Metrics

In face of the previous observations, when comparing the different techniques we mainly use the following metrics: *true positive rate*, *false positive rate*, *accuracy*, and the *area under the ROC curve*. The True Positive Rate (TPR) measures the fraction of positive samples that are correctly identified as such, while the False Positive Rate (FPR) measures the proportion of negative samples erroneously classified as positive. Thus, adversaries will attempt to obtain a high TPR and a low FPR when performing covert traffic classification. Accuracy captures the fraction of correct labels output by the classifier among all predictions, and can be used as a summary of the classification performance since high accuracy implies a high true positive rate and a low false positive rate. The ROC curve plots the TPR against the FPR for the different possible cutout points for classifiers possessing adjustable internal thresholds. The area under the ROC curve (ROC AUC) [16] summarizes this trade-off. While a classifier outputting a random guess has an AUC=0.5, a perfect classifier would achieve an AUC=1, where the optimal point on the ROC curve is FPR=0 and TPR=1.

2.4 Experimental Setup

For conducting our study, we were required to analyze a number of network traces produced by the systems described in Section 2.1. For our testbed, we used two 64-bit Ubuntu 14.04.5 LTS virtual machines (VMs) provisioned with a 2.40GHz Intel Core2 Duo CPU and 8GB of RAM configured in a LAN setting. We used the *v4l2loopback* camera emulator and the *pulseaudio* sound server to feed video and audio to the carrier multimedia applications. The prototypes of the considered systems were obtained from their respective websites [3, 29, 33]. Due to the deprecation of Skype v4.3 and the incompatibility of *v4l2loopback* with the latest Skype v8.x desktop version, we have resorted to Skype for Web. For gathering the traffic samples generated by each system, we captured the network packets produced by the carrier

multimedia streams for a duration of 60 seconds after a given covert channel has been established. The methodology we followed for gathering traffic samples has been commonly used in the literature since it allows for the analysis of the unobservability properties of covert channels while executing in steady-state. Next, we describe the methodology we followed for generating our covert and legitimate traffic datasets.

Facet: For building our covert video dataset, we collected 1000 YouTube videos from the YouTube-curated Top Shared and Liked playlist. The legitimate Skype video dataset consists of 1000 recorded live chat videos available on YouTube. We adapted the Facet prototype to sample three types of Facet transmissions, corresponding to scaling the covert videos on top of legitimate videos by a factor of 50%, 25% and 12.5% – the available prototype represents a proof-of-concept only capable of a (unmorphed) 100% scaling. Then, we gathered 1000 traffic samples for each scaling factor by combining a pair of legitimate and covert videos while following the audio and video morphing techniques detailed in Facet's original description. To emulate legitimate Skype calls, we streamed the media comprising our legitimate Skype video dataset. The resolution of the camera emulator was set to 320x240. For gathering traffic samples, we used each of the available VMs as a Skype peer.

CovertCast: For building our legitimate live-streaming dataset, we crawled 200 live-streams included in the Live YouTube-curated list. Then, we generated 200 CovertCast live-streams by broadcasting several news websites already included in the available CovertCast prototype. The server component, responsible for scraping websites, was executed in one of our VMs and streamed modulated video frames to YouTube. We used a Windows laptop running Google Chrome as a CovertCast client. Each video was streamed with a 1280x720 resolution.

DeltaShaper: We emulated 300 legitimate bi-directional Skype calls by streaming a subset of our legitimate Skype video dataset. We gathered DeltaShaper traffic samples by establishing a DeltaShaper connection between the Skype endpoints installed in both VMs. We gathered data for two DeltaShaper configurations, found to provide traffic analysis resistance guarantees, and which respected the tuple (payload frame area, cell size, number of bits, framerate). These were comprised by the $\langle 320\times240, 8\times8, 6, 1\rangle$ and $\langle 160\times120, 4\times4, 6, 1\rangle$ tuples. Each video was streamed in a 640x480 resolution.

3 Similarity-based Classification

For the purpose of unobservability assessment, multiple similarity functions have been used to feed similarity-based classifiers. This section details the rationale behind each of these functions and how they have been used for the construction of similarity-based classifiers and applied to different multimedia protocol tunneling systems. Then, we conduct a comparative analysis of the performance of each of these classifiers.

3.1 Currently Used Similarity Functions

Next, we introduce the three similarity-based classifiers which have been previously used for evaluating the unobservability of Facet, CovertCast, and DeltaShaper.

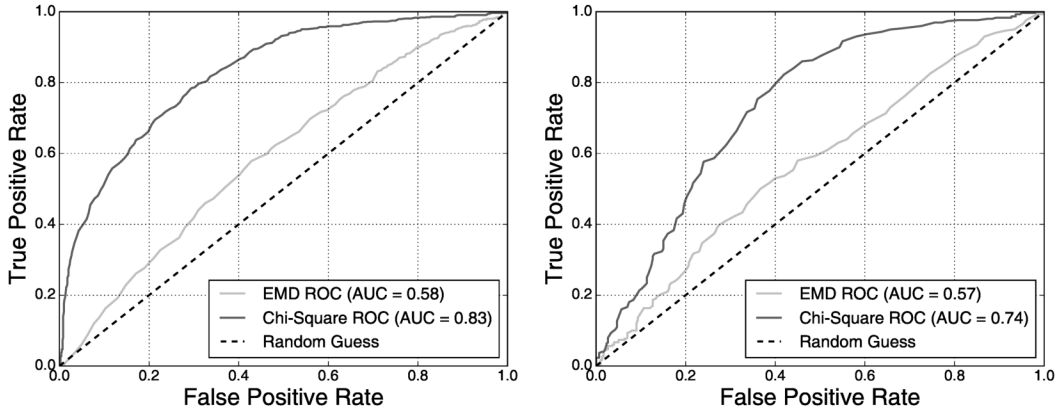
In similarity-based classification [10], labeling is performed by taking into account the pairwise-similarities between the test sample and a set of labeled training samples (or a representative model based on these). In the context of traffic analysis, similarity scores are often obtained from the comparison of the frequency distribution of packet lengths or inter-arrival times of traffic samples.

Pearson's Chi-squared Test (χ^2) [40] tells us whether the distributions of two categorical variables differ significantly from each other, by comparing the observed and expected frequencies of each category. The χ^2 test is used in a classifier adapted for distinguishing Facet traffic [30, 51]. The classifier starts by building two models for legitimate and Facet traffic, respectively, using labeled samples. These models are based upon a selection of the bi-gram distribution of packet lengths, where bi-grams expected to hurt classification performance are identified and discarded. Test samples are compared to each of the models using the χ^2 test. A simpler version of this classifier labels a sample according to the minimum distance obtained when compared against each model. A more sophisticated version of the classifier labels samples according to whether the ratio between the distance to each model surpasses a threshold. An adversary can adjust this threshold for balancing the expected true positive and false positive rates of the classifier.

Kullback-Leibler Divergence (KL) [28] is a measure of relative entropy between two target distributions which is obtained by computing the information lost when trying to approximate one distribution with the other. The KL divergence is used for building a classifier for CovertCast traffic. The classifier aims at distinguishing a set of YouTube videos carrying modulated data from a set of regular YouTube videos through the comparison of the quantized frequency distribution of packet lengths. For each sample in a given set, the classifier computes its KL divergence from every other member in the same set and every member in the other set. Then, the classifier computes a success metric, corresponding to the number of times the KL divergence between a member of one set is more similar to another member of the same set, divided by the total KL divergences that were computed.

Multimedia Protocol Tunneling System	χ^2 Classifier			KL Classifier			EMD Classifier		
	ACC	TPR	TNR	ACC	TPR	TNR	ACC	TPR	TNR
Facet (s=50%)	0.743	0.797	0.689	0.575	0.675	0.476	0.575	0.578	0.572
Facet (s=25%)	0.713	0.795	0.630	0.558	0.615	0.500	0.535	0.827	0.242
Facet (s=12.5%)	0.772	0.793	0.750	0.551	0.596	0.506	0.530	0.793	0.267
DeltaShaper $\langle 320\times240, 8\times8, 6, 1\rangle$	0.690	0.716	0.663	0.546	0.628	0.464	0.567	0.500	0.633
DeltaShaper $\langle 160\times120, 4\times4, 6, 1\rangle$	0.540	0.437	0.650	0.515	0.531	0.500	0.528	0.223	0.833
CovertCast	0.990	1.000	0.980	0.923	0.999	0.846	0.830	0.965	0.695

Table 1: Accuracy, true positive, and true negative rates when detecting covert channels on different multimedia protocol tunneling systems. For the EMD classifier, the threshold value was chosen to be the one providing the highest accuracy, irrespective of the trade-off between the true positive and true negative rates of the classifier.



(a) Facet s=50%

(b) DeltaShaper $\langle 320\times240, 8\times8, 6, 1\rangle$.

Figure 1: ROC curve for the χ^2 and EMD classifiers when identifying Facet and DeltaShaper traffic.

Earth Movers's Distance (EMD) [43] measures the dissimilarity between two distributions, where the distance between single features can be defined in a distance matrix. Informally, this dissimilarity represents the necessary amount of work to turn one probability distribution into the other, where the cost of this transformation translates to the amount of observations moved times the distance defined in the associated distance matrix. The EMD (provided with a unitary distance matrix) is used for comparing the quantized frequency distribution of packet lengths of traffic samples, and is used as basis for building a classifier for DeltaShaper traffic. First, the classifier computes the pairwise EMD between each sample in the dataset and each legitimate sample, recording its average. The intuition is that legitimate samples will exhibit a smaller average EMD. An internal threshold adjusts the trade-off between the true positive and false positive rates of the classifier. For labeling a new sample, the classifier computes the pairwise distance of this sample to each legitimate sample and verifies whether its average EMD surpasses the threshold.

3.2 Main Findings

We now present the main findings of our analysis after assessing the unobservability of each system with all the similarity-based classifiers described above.

1. The claims on the unobservability guarantees of multimedia protocol tunneling systems are intimately tied to the classifier employed in their evaluation. This finding can be illustrated by the numbers in Table 1, which shows the accuracy, true positive and true negative rates obtained by the classifiers described in Section 3.1. For example, when detecting Facet s=50% traffic, we can see that the χ^2 performs relatively well, with an accuracy of 74.3%. Contrastingly, the KL and EMD classifiers attain an accuracy close to random guessing, providing an optimistic estimate on the unobservability of Facet s=50%. The values in Table 1 suggest a similar trend when detecting DeltaShaper and CovertCast traffic. Our results also suggest that χ^2 outperforms all other similarity-based classifiers proposed for the assessment

of unobservability. This can be inferred from the consistently higher accuracy values provided by χ^2.

2. χ^2 produces large false positive rates when classifying Facet and DeltaShaper traffic. Figure 1 depicts the ROC curve of the χ^2 and EMD classifiers when detecting Facet and DeltaShaper traffic. Figure 1a shows that for correctly identifying 90% of all Facet traffic (TPR=90%), with s=50%, an adversary would tag 45% of legitimate connections (45% FPR) as covert traffic, while employing the χ^2 classifier. For identifying 90% of all DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic, the adversary would face an FPR=51%. Thus, even the deployment of the best performing similarity-based classifier results in a large number of misclassifications for legitimate traffic. Misclassifications are further aggravated should an adversary resort to the EMD classifier. Figure 1 confirms that χ^2 performs only fairly in distinguishing covert channels (e.g., AUC=0.83 for Facet s=50%, AUC=0.74 for DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$). We do not show a ROC curve for KL as the classifier is not adjustable by an internal threshold.

3. CovertCast fails to provide unobservability. The results in Table 1 show that the χ^2 classifier can correctly identify all of CovertCast streams while incurring only in a 2% false positive rate. Additionally, the numbers show that the remaining classifiers can correctly identify >96.5% of CovertCast streams, albeit incurring in a larger false positive rate (e.g., EMD: TPR=0.965, FPR=0.305). We conjecture two explanations that may justify the differences beween our results and those published in the original CovertCast paper. Firstly, our results may stem from the use of a dataset which is one order of magnitude larger than the one used for CovertCast evaluation. This increased dataset may more accurately represent the patterns generated by legitimate YouTube streams' traffic and reveal CovertCast activity. Secondly, implementation changes in YouTube may have impacted the unobservability properties provided by hardcoded data modulation parameters, which may in turn be no longer adequate to ensure unobservability.

4 Decision Tree-based Classification

In this section, we depart from the use of similarity-based classifiers for detecting the presence of covert traffic. As it is unpractical to explore all possible machine learning algorithms, we focus our experiments in a subset of algorithms based on decision trees. We have chosen these algorithms due to their ability of handling data in a non-linear fashion, their ability to perform feature selection, and the ease of interpretation of the resulting models. Our results show that this approach is highly effective at detecting covert traffic in the systems under study.

4.1 Selected Classifiers

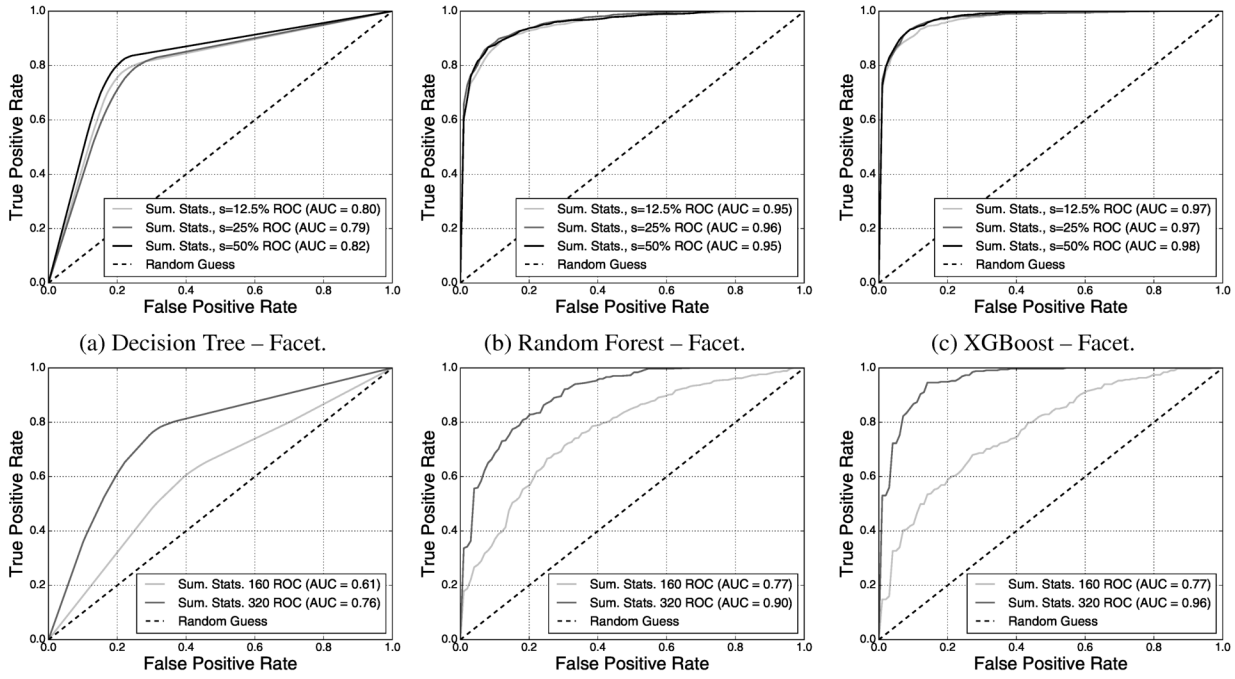
We present a description of the decision-tree based algorithms we have chosen for conducting our experiments:

Decision Trees [41] build a model in the form of a tree structure, where each tree node is either a decision or leaf node, representing a branch or a label, respectively. Decision nodes split the current branch by an attribute. A splitting attribute is commonly chosen according to its expected information gain, i.e. the expected reduction in entropy caused by choosing the attribute for a split. The importance of each particular attribute can be assessed by analyzing the tree structure, where nodes closer to the root have a higher importance than those down the tree. Despite its simple interpretation, decision trees can result in complex models unable to generalize well or can build unstable models due to the presence of large numbers of correlated features. A popular way to mitigate such disadvantages is to use decision tree ensembles.

Random Forests [6] are an ensemble learning method, where a label is predicted by performing a majority vote over the output of multiple decisions trees. To prevent overfitting, Random Forests introduce variance in the model through *bootstrap aggregation*, i.e. each tree is trained using a random sample (with replacement) of the training set. Additionally, Random Forests select random attributes of the feature set when building each tree, a technique named *feature bagging*. One method for assessing the importance of an attribute is to average its information gain across all trees in the ensemble.

eXtreme Gradient Boosting (XGBoost) [9] is another technique for building a model based on an ensemble of decision trees; it relies on a technique known as *gradient tree boosting*. XGBoost starts by building a shallow decision tree (i.e., a weak learner). In each step, XGBoost creates a new tree which optimizes the predictions performed by trees in earlier stages. XGBoost benefits from a regularized model formalization to control overfitting. The importance of individual attributes can be computed in a similar fashion to that of Random Forests. We find the use of XGBoost to be promising among a large pool of classification algorithms. In fact, XGBoost has played a central role on multiple winning solutions for recent data mining competitions, spawning multiple domains, such as the KDD Cup 2016 [12, 44]

The next sections detail our experiments for evaluating the unobservability of Facet and DeltaShaper with the decision tree-based classifiers enumerated above. In our experiments we have used two distinct sets of features: summary statistics and quantized packet lengths. We omit a discussion over CovertCast, as we have found that all of these techniques can identify its covert traffic with a negligible false positive rate.



(a) Decision Tree – Facet. (b) Random Forest – Facet. (c) XGBoost – Facet.

(d) Decision Tree – DeltaShaper. (e) Random Forest – DeltaShaper. (f) XGBoost – DeltaShaper.

Figure 2: ROC curves for decision tree-based classifiers when classifying Facet and DeltaShaper traffic using Feature Set 1 (summary statistics).

4.2 Feature Set 1: Summary Statistics

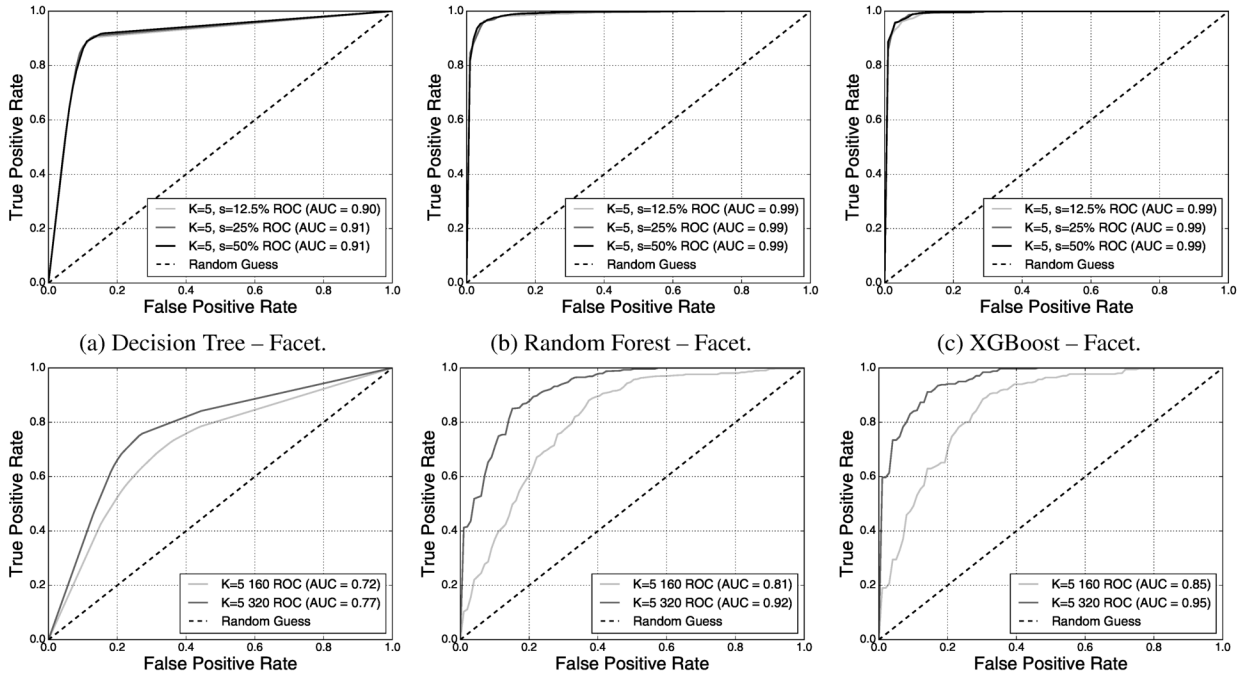
The collection of encrypted traffic provides an adversary with two main sources of data for extracting features necessary for the detection of covert channels: a timeseries of packet lengths, and a timeseries of packet inter-arrival times. Our first set of features comprises a collection of summary statistics computed over the network traces of legitimate and covert traffic. This is a prevalent approach at generating features for the problem of encrypted traffic fingerprinting [22, 38, 49]. Such set of features has not been previously applied in the detection of covert channels generated by multimedia protocol tunneling.

As for the choice of summary statistics, we compute multiple descriptive statistics for the ingress/ egress packet flows of a connection as a whole, as well as for ingress/ egress traffic individually. This feature set includes simple descriptive statistics over the packet length and inter-arrival time timeseries – such as maximum, minimum, mean, and percentiles – as well as higher-order statistics like the skew or kurtosis of these timeseries. We also consider burst behavior [1], where a burst is a sequence of consecutive packets transmitted along the same direction of a given connection. A total of 166 features are used for training our classifiers. Due to space constraints, we relegate a full listing of the summary statistics we have considered to the appendix.

Next, we present our main findings after attempting to detect multimedia protocol tunneling covert channels using the decision-tree based classifiers we have described, while feeding them with our collection of summary statistics. We report the performance of each classifier over 10-fold cross-validation.

1. The use of Random Forest/ XGBoost, used in tandem with summary statistics, largely undermines the unobservability claims of state-of-the-art multimedia protocol tunneling systems. Figure 2 shows the ROC curve for our decision tree-based classifiers when detecting Facet and DeltaShaper traffic resorting to summary statistic features (ST). Random Forest – ST exhibits a minimum AUC=0.95 when classifying all configurations of Facet traffic, while XGBoost – ST exhibits a minimum AUC=0.97. When compared to XGBoost – ST, the χ^2 classifier attains a maximum AUC=0.85. For DeltaShaper traffic, XGBoost – ST attains an AUC which is 0.22 larger for both DeltaShaper configurations, when compared to that obtained by the χ^2 classifier.

2. It is possible to flag a vast majority of covert channels with a very small number of false positives. An adversary that aims at flagging at least 90% of all Facet s=50% connections incurs in a 14.1% FPR when resorting to Random Forest – ST, and a FPR as short as 7.1% when resorting to XGBoost – ST. To flag at least 70% of the same kind of traffic, XGBoost – ST incurs in a FPR of only 1%. In comparison, Figure 1a shows that



(a) Decision Tree – Facet. (b) Random Forest – Facet. (c) XGBoost – Facet.

(d) Decision Tree – DeltaShaper. (e) Random Forest – DeltaShaper. (f) XGBoost – DeltaShaper.

Figure 3: ROC curves for decision tree-based classifiers when classifying Facet and DeltaShaper traffic using Feature Set 2 (quantized frequency distribution of packet lengths).

for correctly identifying just 70% of Facet s=50% traffic when resorting to the χ^2 classifier, an adversary would face an alarming 21.5% FPR. The situation is similar for an adversary wishing to flag 90% of DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic. For flagging 90% of this kind of traffic, Random Forest – ST incurs in a 30.3% FPR and XGBoost – ST incurs in a 12.1% FPR. To flag 70% of the same kind of traffic, XGBoost – ST incurs in a FPR of 4%. Flagging just 70% of this kind of traffic with the χ^2 classifier would amount to a 32.2% FPR.

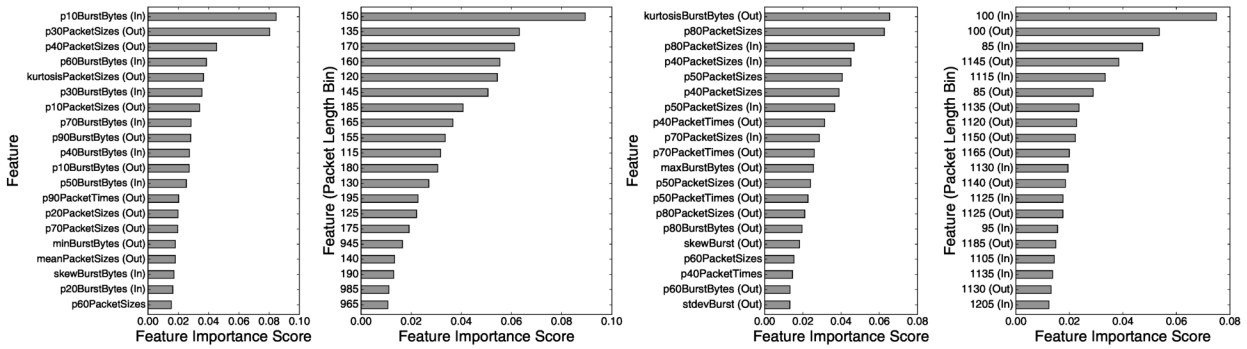
4.3 Feature Set 2: Quantized PLs

An alternative feature set is comprised of the quantized frequency distribution of packet lengths, where each K size bin acts as an individual feature. While this feature set is akin to that previously used in KL and EMD similarity-based classifiers, we process these features in a fundamentally different way. In particular, the similarity-based classifiers output a distance score based on the overall difference of the packet lengths frequency distribution, while failing to adjust this score according to the importance of relevant regions of the feature space. Informally, they risk to dilute the greater discriminating power of a given feature among that of possibly irrelevant features [35]. We aim at exploiting the different relevance of particular ranges of the feature space by feeding this feature set to decision tree-based classifiers.

In terms of feature sets, for Facet, we take as features the quantized frequency distribution of packet lengths for the flow carrying covert data. We use K=5 as we have experimentally verified that the classification performance of our decision tree-based algorithms benefit from a fine-grained quantization. As for DeltaShaper, and due to the system's bidirectionality, we use the quantized frequency distribution of packet sizes flowing in both directions. Here, we also apply a quantization with K=5. Note that the evaluation performed with the similarity-based classifiers described in Section 3 also considers the same selection on the direction of traffic flows to analyze.

Next, we describe our findings after attempting to identify covert traffic with such feature sets. Figure 3 shows the ROC curve for our decision tree-based classifiers when detecting Facet and DeltaShaper traffic resorting to quantized packet lengths as features (PL).

1. Quantized packet lengths outperform the use of summary statistics. In general, the AUC obtained by our decision-tree based classifiers is comparable or superior to the AUC obtained by the same classifiers when making use of summary statistics. Both Random Forest - PL and XGBoost - PL obtain an AUC=0.99 when identifying Facet traffic. This represents a maximum improvement of 0.04 over Random Forest – ST and



(a) ST - Facet. (b) PL - Facet. (c) ST - DeltaShaper. (d) PL - DeltaShaper.

Figure 4: Top 20 most important features when classifying Facet s=50% and DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic, as calculated by XGBoost. We report the mean score of each feature across all 10 cross-validation folds.

0.02 over XGBoost – ST. While Decision Tree - PL attains a maximum AUC=0.91, it is still short of the maximum AUC attained by Random Forest – ST. This trend is similar in the classification of DeltaShaper traffic, where the AUC obtained by Decision Tree - PL is also inferior to that of tree ensembles. The detection of $\langle 160 \times 120, 4 \times 4, 6, 1 \rangle$ DeltaShaper traffic benefits the most from packet length features, where XGBoost - PL attains an AUC=0.85, 0.08 larger than that obtained by XGBoost – ST. Interestingly, the detection of $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ DeltaShaper traffic is better performed by XGBoost – ST, albeit by a slight improvement of 0.01 over the AUC of XGBoost - PL.

4.4 Feature Importance

The above set of experiments allowed us to implicitly identify which features are more important to distinguish between two classes of traffic. Figure 4a shows the top 20 most important summary statistics for detecting Facet traffic s=50%, as reported by the XGBoost algorithm. Figure 4b summarizes the 20 most important quantized ranges of packet lengths. The features annotated with "Out" correspond to those generated by the packet flow directed towards the client (carrying the covert payload), while the features annotated with "In" correspond to the packet flow directed towards the Facet server.

Figure 4c depicts the top 20 most important summary statistics for detecting DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic, as reported by XGBoost. Similarly, Figure 4d depicts the most important quantized ranges of packet lengths for detecting the same kind of traffic. Each feature is annotated with "Out" or "In", depending on the particular Skype peer originating covert traffic. We note that both peers generate covert traffic simultaneously due to DeltaShaper's bidirectionality. Below, we discuss the main findings of our analysis.

1. Facet is more vulnerable to analysis based on packet lengths and burst behavior. Figure 4a shows that Facet detection is driven by features related to the packet lengths and the burst behavior of the connection, whereas packet timing does not contribute as much. An interesting observation is that the majority of packet bursts features considered important for classification are those included in the flow directed towards the Facet server, which carries no covert data. This fact suggests that Skype flows exhibit some degree of co-dependency and that both flows provide useful information for distinguishing between legitimate and covert transmissions. Features included in the top 10, and that directly concern the length of packets, index summary statistics from the flow carrying covert data. This suggests that the flow carrying covert data is the prime target for inspection when analyzing packet lengths. Additionally, packet lengths comprehended between the 10th and 40th percentiles, amounting to packets with a mean length comprehended between 138 and 213 bytes, have a superior discriminating power among other packet sizes. XGBoost ranks 123 of the 166 features with a non-zero importance score.

2. DeltaShaper is more vulnerable to analysis based packet lengths. Figure 4c shows that 10 of the most important features for detecting DeltaShaper regard descriptive statistics of packet lengths. In particular, 7 out of the top 10 most important features for identifying DeltaShaper traffic are related to the length of transmitted packets. Contrary to Facet, these features include a mixture of traffic originating in different peers, which is expected according to the bidirectionality of the covert channel. We find the most influential packet lengths to be within the range of the 40th and 80th percentiles, amounting to packets with a mean length comprehended between 1026-1180 bytes. XGBoost ranks 132 of the 166 features with an importance score larger than zero.

3. Facet covert channels can be spotted by looking for packets with a length comprehended between 115-195 bytes. Figure 4b not only shows that the most important bin corresponds to that by the packets which length is close to 150, but also that the top 10 features are dominated by packets which lengths are in the range of 115 to 195 bytes. This result concurs with our previous observation, where the most important percentiles of packet lengths focused packets with a mean length between 137 and 200 bytes. This observation is also true when detecting Facet s=$\{12.5\%,25\%\}$ traffic. This finding suggests that the major factor leading to the distinguishing of Facet traffic concerns the packets carrying audio, which are typically located in the range between 100 and 200 bytes [37]. Additionally, we can observe that some of the least important features included in the top 20 for identifying Facet s = 50% flows include packets with a length between 945-985 bytes. This result hints that larger areas dedicated to video payload translate into packet-level modifications in a higher range of the feature space. Additionally, XGBoost ranks only 175 out of 300 features with a non-zero importance score, suggesting that only approximately half of the quantized packet length bins contribute for the discrimination of Facet traffic.

4. DeltaShaper covert channels can be spotted by looking for packets with a length between 85-100 and 1105-1205 bytes. Figure 4d shows that the two most important features for identifying DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic correspond to the packets which size is close to 100 bytes (flowing in both directions). The top 20 features are dominated by packet length bins in the range from 85-100 and 1105-1205 bytes, suggesting that DeltaShaper data modulation markedly affects two distinct regions of the feature space. The region including larger packets roughly overlaps the mean length of the packets included in the most important percentiles of our analysis of summary statistics. Considering that DeltaShaper's covert data embedding procedure specifically targets the video layer of Skype calls, this finding suggests that such modulation largely affects larger packets of the connection. When classifying DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic, XGBoost ranks 253 out of 600 features with a non-zero importance score.

The most important features for detecting DeltaShaper $\langle 160 \times 120, 4 \times 4, 6, 1 \rangle$ traffic largely overlap the two feature set regions already reported. However, we verify that the region including larger packet lengths was significantly expanded, including bins representing packets with a size within the range of 885-1200 bytes.

4.5 Alternative Dataset Evaluation

We have constructed and handled our dataset by following the same methodology adopted by previous works under study. However, this methodology may raise a few concerns. In particular, the covert streams (positive class) have been produced using the available legitimate videos (negative class), which may introduce some form of correlation among classes. Furthermore, this methodology generates a 1:1 ratio of positive to negative classes, which may be unrealistic if covert streams are a minority among the traffic found in the wild. Thus, one may wonder how accurate is our classifier if: i) the positive class is no longer correlated with the negative class during testing; ii) the positive-to-negative sample ratio is low during testing. To validate the effectiveness of our approach, we performed two additional experiments.

First, we performed an experiment which removed the correlations between the positive and negative classes. We split our legitimate traffic dataset in half, using only one half as legitimate samples. Then, for creating our covert video dataset, we selected those covert videos which embed modulated data in the legitimate videos out of our reduced legitimate traffic dataset. We then used XGBoost to build a model through 10-fold cross-validation. To prevent the fitting of results to a particular choice of the initial legitimate samples, we repeated the process 10 times while randomly choosing such samples.

Second, we performed an experiment where we keep the positive-to-negative sample ratio low during testing. We split our data in training / testing sets in a 70 / 30 proportion, and where we kept the training set ratio as 1:1, and keep the positive to negative ratio of the testing set to 1:100. To prevent the fitting of results to a particular split of the data, we randomly choose each set 10 times.

The results of our additional experiments suggest that possible correlations among training and testing data, as well as sample ratios, do not limit the accuracy of our approach. For our first experiment, XGBoost obtained an AUC=0.94 for DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic (only 0.01 less than the results reported in Section 4.3), and an AUC=0.99 for traffic pertaining to Facet s=50% configuration. As for the second experiment, XGBoost was able to correctly identify 90% of Facet s=50% traffic with an FPR of only 2%, while it was able to identify 90% of DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic with an FPR of 18% (only 4% larger).

4.6 Practical Considerations

This section details several practical considerations which may be useful to an adversary considering the use of decision tree classifiers for the detection of covert channels. The following results reflect processing time in a VM configuration akin to that described in Section 2.4.

Feature extraction. The extraction of quantized packet length bins from a 60 second Facet network trace amounts to an average of 0.33s per sample. Generat-

System	Feature Set	Memory (kB)	Storage (kB)
Facet	Summary Statistics (ST)	1.3	1.8
	Packet Lengths (PL)	2.4	1.0
DeltaShaper	Summary Statistics (ST)	1.3	1.9
	Packet Lengths (PL)	4.8	2.0

Table 2: Memory and storage requirements for a single Facet record using different feature sets. We report storage requirements for holding data in raw ASCII text.

System	Classifier	Model Building (s)	Prediction (μs)
Facet	Decision Tree	0.27	40
	Random Forest	1.45	15000
	XGBoost	0.41	180
DeltaShaper	Decision Tree	0.13	90
	Random Forest	0.86	16000
	XGBoost	0.38	350

Table 3: Model building time and time for individual predictions for Facet s=50% and DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic, using quantized packet lengths (PL). Model building time is the average of 10 folds.

ing summary statistics describing the same type of traffic flow amounts to an average of 0.44s per sample. This result indicates that an adversary can quickly generate feature vectors for conducting subsequent classification.

Memory and storage requirements. Table 2 depicts the memory and storage requirements for holding a single Facet or DeltaShaper sample. In our Python implementation, a NumPy [47] array storing the quantized packet lengths describing a Facet sample (300 attributes) occupies 2.4kB of memory per sample. In comparison, an array containing the bi-grams required by the χ^2 classifier occupy a total of 45kB per sample. The numbers in Table 2 suggest that an adversary can efficiently store and process large datasets. As an example, storing 1 million Facet quantized packet lengths feature vectors in a raw ASCII text file would only occupy approximately 1GB of disk space. Storing summary statistics in raw ASCII text would occupy nearly twofold the space due to the characters required to represent floating-point precision.

Model building and classification speed. Table 3 depicts the average training time of our classifiers, as well as the average time to output a prediction. Building a Decision Tree - PL for identifying Facet traffic takes an average of 0.27s. For an ensemble composed of 100 trees, Random Forest - PL and XGBoost – PL models are built in 1.45s and 0.41s, respectively. Moreover, the average classification time for an individual sample is 180μs for XGBoost – PL. XGBoost is not only more accurate but also trains faster and exhibits a faster classification speed than Random Forest. This relation is also present when classifying DeltaShaper traffic. These results stress the fact that an adversary would benefit from using XGBoost to detect multimedia protocol tunneling covert channels.

System	1s	5s	10s	30s	60s
Facet	0.81	0.92	0.96	0.99	0.99
DeltaShaper	0.75	0.88	0.93	0.95	0.95

Table 4: AUC of XGBoost – PL when classifying Facet s=50% and DeltaShaper $\langle 320 \times 240, 8 \times 8, 6, 1 \rangle$ traffic for varying traffic collection time windows.

Generalization ability of the classifiers. A classifier with good generalization ability is able to perform correct predictions for previously unseen data. Albeit the AUC obtained by our decision tree-based classifiers suggests that these can generalize well, we further assess their classification performance when training data is severely limited. We split our data in two 10 / 90 training and testing sets, and report the mean AUC obtained by the classifier after repeating this process 10 times while randomly choosing the samples making part of each set. In this setting, when classifying Facet s=50%, XGBoost - PL attains an AUC=0.98, only 0.01 short of that obtained after 10x cross-validation. For DeltaShaper $\langle 160 \times 120, 4 \times 4, 6, 1 \rangle$ traffic, XGBoost - PL attains an AUC 0.1 smaller than their 10x cross-validation counterpart. These results suggest that an adversary can build accurate decision tree-based classifiers for detecting covert traffic while resorting to a small sample of data.

Impact of network traces collection time. Table 4 depicts the AUC obtained by XGBoost – PL when detecting different types of covert traffic for varying time-spans of traffic flows collection. Results show that capturing traffic by 30s is enough for attaining the same classification performance achieved in our initial experiments, which admitted 60s traffic captures. The numbers in Table 4 also show that classification performance decreases monotonically for traffic collections fewer than 30s, suggesting that the inspection of at least 30s of video traffic provides the adversary with sufficient data for identifying covert traffic flows with low false positives.

5 Beyond Supervised Anomaly Detection

While decision tree-based classifiers show promising results for the detection of multimedia protocol tunneling covert channels, they require the adversary to obtain a labeled dataset, including both legitimate and covert traffic. This usually requires the adversary to have a unlimited access to a particular multimedia protocol tunneling tool with which it may generate covert traffic samples. However, even if an adversary, for instance a censor, would have an expedite access to these tools [19], it is interesting to understand if detection is possible without

this knowledge. Note that covert channels may also be used by organized criminals that can succeed in delaying the dissemination of such tools. Secondly, albeit the adversary is assumed to possess a given tool, it is expected to spend a non-negligible time in synthesizing covert data samples for building a model. Overcoming such challenges opens a timeframe where the covert traffic generated by a given system would remain undetected.

This section explores alternative approaches at covert traffic detection in the absence of a fully labeled dataset.

5.1 Selected Anomaly Detection Methods

This section starts by describing several anomaly detection techniques which could be of interest for an adversary aiming at detecting covert traffic when it is deprived of labeled anomalies. First, we describe OCSVMs and autoencoders, two well-known approaches for anomaly detection, which are based on representational models of legitimate data and thus disregard the need of labeled anomaly data [50]. Then, we explore Isolation Forest, a competitive approach at unsupervised anomaly detection which does not require labeled data [4, 8, 26].

One-class SVMs [45] define a decision boundary between normal samples and anomalies by fitting a function around normal samples during training. OCSVMs attempt to find the maximal margin hyperplane which separates the normal data from the origin, which is treated as the single member of a second class. If data cannot be easily separated by a linear function, OCSVMs project the original feature space into a new feature space through the use of kernel functions, introducing non-linearity in the model. New data samples falling outside the decision boundary are considered anomalies.

Autoencoders [32] are a type of artificial neural networks which can approximate the identity function through a compressed representation of its inputs, forcing the algorithm to learn underlying structures in data. The ability to reconstruct inputs allows us to have a generative model of the training data. An autoencoder can be repurposed for anomaly detection by comparing the reconstruction error of training inputs with normal and anomalous data, where the latter is assumed to be larger.

Isolation Forest [31] performs outlier detection by isolating anomalous samples. To isolate a sample, the algorithm starts by selecting a random feature and selects a split between its minimum and maximum values. This process continues recursively until the considered sample is isolated. Recursive partitioning is represented by a tree, where the number of partitions required to isolate a sample corresponds to the length of the path traversed from the root node to a leaf. The Isolation Forest is built by combining a number of isolation trees split on different attributes. Anomalies are expected to exhibit a smaller average path length than that of normal samples.

Hyperparameters. The classification performance of the above algorithms depends upon the choice of hyperparameters, i.e., parameters whose value must be set prior to the execution of the algorithm. The optimality of such parameters is intrinsically dependent on the dataset and tipically requires cross-validation with labeled anomalous data [56]. However, we are interested in assessing the average classification performance that an adversary would be able to achieve using such algorithms – albeit the adversary would be unable to find the optimal hyperparameter configuration for an algorithm, sub-optimal parameterizations may still provide the adversary with accurate traffic classifiers. To this end, we conduct a search over a space of parameters for the above algorithms and collect the maximum and average AUC obtained when classifying Facet and DeltaShaper traffic.

For OCSVM, we perform a grid search on the space of ν and γ. We also build a shallow autoencoder containing one hidden layer between the input and its compressed representation, and between the compressed representation and the output layer. We conduct a grid search over the number of units populating each of these layers. As for Isolation Forest, we conduct a search over the number of trees composing the ensemble, as well as the number of samples for training each individual tree.

Experimental settings. For OCSVM and autoencoder, we use 90% of all labeled legitimate samples to learn the models. The remaining 10% legitimate samples are combined with 10% of a given covert traffic configuration's samples for creating a balanced testing set. For evaluating the model's performance, we compare each label output by the model with the ground truth. To prevent the fitting of results to a particular split of the data, we repeat this process 10 times while randomly choosing the samples making part of the training / testing sets. For Isolation Forest, we create balanced training and testing sets in a 90 / 10 proportion. The model's performance is evaluated following the same above procedure.

Our results reflect the use of the feature set based on the frequency distribution of packet lengths, with $K = 5$, as it was the one found to provide the highest AUC.

5.2 Main Findings

Table 5 depicts the maximum and average AUC obtained when identifying Facet and DeltaShaper traffic when using OCSVM, our autoencoder, and Isolation Forest. Next, we present our main findings.

1. OCSVMs possess a limited capability for correctly identifying covert traffic. This finding is supported by the fact that OCSVM attains an average

Multimedia Protocol Tunneling System	OCSVM		Autoencoder		Isolation Forest	
	Max AUC	Avg AUC	Max AUC	Avg AUC	Max AUC	Avg AUC
Facet (s=50%)	0.631	0.576	0.702	0.638	0.561	0.551
Facet (s=25%)	0.629	0.580	0.700	0.650	0.528	0.519
Facet (s=12.5%)	0.639	0.584	0.706	0.647	0.536	0.520
DeltaShaper $\langle 320\times240, 8\times8, 6, 1\rangle$	0.567	0.531	0.662	0.574	0.580	0.557
DeltaShaper $\langle 160\times120, 4\times4, 6, 1\rangle$	0.548	0.518	0.576	0.544	0.553	0.532

Table 5: Maximum and average AUC of OCSVM, Autoencoder and Isolation Forest when classifying Facet and DeltaShaper traffic. Search (min, max, step): OCSVM (ν(0.1, 1, +0.1), γ(0.01, 1, +0.01)); Autoencoder (hidden_layers(4,512,*2), compressed_representation(4,512,*2), learning_rate[0.001,0.01], epochs[1000]); Isolation Forest (n_trees(50,200,*2), n_samples(64,512,*2))

AUC between 0.576 and 0.584 when detecting Facet traffic, and between 0.518 and 0.531 when detecting DeltaShaper traffic. Moreover, OCSVM achieves a maximum AUC=0.639 when classifying Facet s=12.5% traffic. This suggests that OCSVM achieves a poor classification performance, even after a search for optimal hyperparameters. Thus, from an adversary's point of view, a semi-supervised model based on OCSVMs shows little promise for performing the triage of covert traffic.

2. Autoencoders show promising results for the identification of covert traffic. The numbers in Table 5 show that our autoencoder achieves, in average, a higher or comparable AUC than the maximum AUC obtained by OCSVM when classifying Facet or DeltaShaper traffic. The choice of parameters for our autoencoder benefits its maximum AUC. For instance, a better parameterization of the autoencoder translates into a maximum AUC=0.662 when classifying DeltaShaper traffic, approximately 0.1 higher than the average reported value for the same configuration. While an adversary making use of a classifier which exhibits an AUC=0.662 would sustain a large amount of false positives when attempting to detect covert traffic, we note that the obtained results have a wide margin of improvement. In particular, we use a rather shallow autoencoder structure for investigating the classification performance of this algorithm. For instance, it is possible that autoencoders with more sophisticated structures [55] may drive further improvements in classification accuracy.

3. An adversary has no advantage in using Isolation Forest for detecting covert traffic. The results in Table 5 show that the prediction output of Isolation Forest is close to random guessing when attempting to identify covert traffic. For Facet traffic, Isolation Forest obtains an average AUC between 0.519 and 0.551 across all steganography factors. When classifying DeltaShaper traffic, the average AUC sits on 0.532 and 0.557 for different encoding configurations. A closer observation of the confusion matrix reveals that Isolation Forest labels few traffic samples as anomalies. Informally, this observation suggests that anomalies are able to conceal their presence in the dataset in such a way that the number of partitions required to isolate them is similar to the number of partitions needed to isolate legitimate samples.

6 Discussion

We now discuss several relevant findings from our study.

Multimedia protocol tunneling. The outcomes of the experimental study conducted in Section 4 unveil that the unobservability claims of existing multimedia protocol tunneling systems were flawed. However, it is worth noticing that the vulnerability of such systems to supervised ML techniques, particularly decision tree-based algorithms, does not imply that multimedia protocol tunneling, as an approach, is fundamentally inviable. Our findings suggest that correctly detecting covert channels built with conservative data modulation schemes (e.g., DeltaShaper $\langle 160\times120, 4\times4, 6, 1\rangle$) while sustaining low FPR still represents a challenge for adversaries. Additionally, we provide fine-grained details about the network behavior of currently deployed multimedia protocol tunneling tools which may be used for the construction of more robust implementations.

Legitimate traffic dataset. Adversaries face the non-trivial challenge of building a dataset which faithfully represents legitimate traffic. A naïve solution for building such a dataset would be for an adversary to take advantage of its privileged position in the network and collect all data originated by a given multimedia protocol. However, the very existence of multimedia protocol tunneling tools makes it hard for an adversary to know, before-hand, which data samples correspond either to legitimate or covert traffic. It is possible that covert data samples pollute the legitimate traffic model and bias the decisions of a classifier trained in such data [55]. A different alternative is the typical approach followed in the literature (and in our work), where datasets are synthe-

sized by transmitting the media expected to be sent in such channels. However, such an approximation may fail to capture the underlying distribution of data in the wild.

7 Related Work

Freewave [25] was the first system designed to embed covert data in multimedia protocols through the modulation of audio signals sent through VoIP streams. However, a simple statistical analysis of traffic patterns conducted by Geddes et al. [20] showed that FreeWave could be trivially detected by an adversary. Recent multimedia protocol tunneling systems such as Facet [30], CovertCast [34], and DeltaShaper [2] introduced new techniques for modulating data while striving to preserve the unobservability of the generated covert channels.

As noted earlier in the text, previous unobservability assessments performed on state-of-the-art multimedia protocol tunneling systems which rely on traffic classification make use of similarity-based classifiers. To the best of our knowledge, there is a limited body of work employing other machine learning techniques for the detection of covert channels in the Internet. Wang et al. [48] have resorted to decision tree-based classifiers to identify traffic flowing through Tor bridges. Their results have shown that this approach was promising for the identification of traffic obfuscated through domain fronting [18]. In our work, we perform the first systematic study of the unobservability of state-of-the-art multimedia protocol tunneling systems and find that such techniques are also effective for the detection of these covert channels.

Related to the problem of covert channel detection is the problem of creating fingerprints for encrypted traffic. Particularly, the fingerprinting of websites accessed through Tor [11] is an important research topic [1, 22, 39, 42, 49]. Multiple works dwell on creating fingerprints for encrypted traffic using different combinations of features and classifiers, for instance, Schuster et al. [46] have designed an attack which enables a passive observer to fingerprint YouTube video streams. However, fingerprinting is fundamentally different from covert channel detection: we do not aim to unequivocally fingerprint a given media according to its traffic pattern, but to distinguish two broader classes of media which may or may not carry covert data. It is unclear how fingerprinting techniques can be adapted to our purpose.

In this paper we have focused on covert channels based on multimedia protocol tunneling [2, 25, 30, 34], a popular approach at protocol tunneling. Other tunneling approaches have been attempted, including SWEET [57], CloudTransport [7], Castle [21], and *meek* [18]. It is worth mentioning that alternative approaches to build covert channels have been attempted in the past, such as protocol obfuscation [52]. However, obfuscation based on randomizing traffic fails in the presence of protocol whitelisting and is vulnerable to entropy analysis [48]. With protocol imitation, covert traffic is manipulated to mimic the behavior of protocols allowed across a censor's border [13, 14, 36]. Alas, the faithful imitation of all behaviors of a protocol behavior is a complex undertaking which lays protocol imitation systems prone to multiple network attacks [20, 23].

Finally, we would like to stress that although censorship circumvention is one of the main (and most noble) uses of covert channels, this type of channels can serve multiple purposes. Our work concentrates on covert channel detection and not on censorship circumvention *per se*. In fact, there are techniques to evade censorship, such as refraction networking [5, 15, 24, 27, 53, 54], which incorporates censorship resistance mechanisms in the network, rather than at end-hosts, that do not depend exclusively on the use of covert channels.

8 Conclusions

In this paper, we performed an extensive analysis over the unobservability evaluation of multimedia protocol tunneling systems. We proposed a novel method for assessing the unobservability of these systems, based on decision trees, which largely defies previous unobservability claims. Our work further explored the application of semi-supervised and unsupervised anomaly detection techniques in the same context. Our results indicate that an adversary is required to possess labeled data for performing an effective detection of covert channels.

9 Acknowledgments

This work was partially supported by national funds through Instituto Superior Técnico, Universidade de Lisboa, and Fundação para a Ciência e a Tecnologia (FCT) via projects PTDC/EEI-SCR/1741/2014, SFRH/BSAB/135236/2017, and UID/CEC/50021/2013.

References

[1] AL-NAAMI, K., CHANDRA, S., MUSTAFA, A., KHAN, L., LIN, Z., HAMLEN, K., AND THURAISINGHAM, B. Adaptive encrypted traffic fingerprinting with bi-directional dependence. In *Proceedings of the 32nd Annual Conference on Computer Security Applications* (Los Angeles, CA, USA, 2016), pp. 177–188.

[2] BARRADAS, D., SANTOS, N., AND RODRIGUES, L. Deltashaper: Enabling unobservable censorship-resistant tcp tunneling over videoconferencing streams. In *Proceedings on Privacy Enhancing Technologies* (Minneapolis, MN, USA, 2017), vol. 2017(4), pp. 5–22.

[3] BARRADAS, D., SANTOS, N., AND RODRIGUES, L. DeltaShaper prototype. `https://dmbb.github.io/DeltaShaper/`, 2017. Last Accessed: 2018-02-05.

[4] BIGML. Which algorithm does BigML use for Anomaly Detection? https://support.bigml.com/hc/en-us/articles/206746259. Last Accessed: 2018-01-16.

[5] BOCOVICH, C., AND GOLDBERG, I. Slitheen: Perfectly imitated decoy routing through traffic replacement. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (Vienna, Austria, 2016), pp. 1702–1714.

[6] BREIMAN, L. Random forests. *Machine learning 45*, 1 (2001), 5–32.

[7] BRUBAKER, C., HOUMANSADR, A., AND SHMATIKOV, V. Cloudtransport: Using cloud storage for censorship-resistant networking. In *Privacy Enhancing Technologies*, E. De Cristofaro and S. Murdoch, Eds., vol. 8555 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 1–20.

[8] CALHEIROS, R. N., RAMAMOHANARAO, K., BUYYA, R., LECKIE, C., AND VERSTEEG, S. On the effectiveness of isolation-based anomaly detection in cloud data centers. *Concurrency and Computation: Practice and Experience 29*, 18 (2017).

[9] CHEN, T., AND GUESTRIN, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd Conference on Knowledge Discovery and Data Mining* (San Francisco, CA, USA, 2016), ACM, pp. 785–794.

[10] CHEN, Y., GARCIA, E. K., GUPTA, M. R., RAHIMI, A., AND CAZZANTI, L. Similarity-based classification: Concepts and algorithms. *Journal of Machine Learning Research 10*, Mar (2009), 747–776.

[11] DINGLEDINE, R., MATHEWSON, N., AND SYVERSON, P. Tor: The second-generation onion router. In *Proceedings of the 13th Conference on USENIX Security Symposium* (San Diego, CA, USA, 2004).

[12] DISTRIBUTED (DEEP) MACHINE LEARNING COMMUNITY. https://github.com/dmlc/xgboost/tree/master/demo#machine-learning-challenge-winning-solutions, 2018. Accessed: 2018-05-31.

[13] DYER, K. P., COULL, S. E., RISTENPART, T., AND SHRIMPTON, T. Protocol misidentification made easy with format-transforming encryption. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security* (Berlin, Germany, 2013), pp. 61–72.

[14] DYER, K. P., COULL, S. E., AND SHRIMPTON, T. Marionette: A programmable network-traffic obfuscation system. In *Proceedings of the 24th USENIX Conference on Security Symposium* (Washington, D.C., USA, 2015), pp. 367–382.

[15] ELLARD, D., JONES, C., MANFREDI, V., STRAYER, W. T., THAPA, B., VAN WELIE, M., AND JACKSON, A. Rebound: Decoy routing on asymmetric routes via error messages. In *Proceedings of the 2015 IEEE 40th Conference on Local Computer Networks (LCN)* (Clearwater Beach, FL, USA, 2015), pp. 91–99.

[16] FAWCETT, T. Roc graphs: Notes and practical considerations for researchers. *Machine Learning 31* (01 2004), 1–38.

[17] FIFIELD, D. *Threat modeling and circumvention of Internet censorship*. PhD thesis, EECS Department, University of California, Berkeley, 2017.

[18] FIFIELD, D., LAN, C., HYNES, R., WEGMANN, P., AND PAXSON, V. Blocking-resistant communication through domain fronting. In *Proceedings on Privacy Enhancing Technologies 2015.2* (Philadelphia, PA, USA, 2015), pp. 46–64.

[19] FIFIELD, D., AND TSAI, L. Censors' delay in blocking circumvention proxies. In *Proceedings of the 6th USENIX Workshop on Free and Open Communications on the Internet* (Austin, TX, USA, 2016).

[20] GEDDES, J., SCHUCHARD, M., AND HOPPER, N. Cover your acks: Pitfalls of covert channel censorship circumvention. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security* (Berlin, Germany, 2013), pp. 361–372.

[21] HAHN, B., NITHYANAND, R., GILL, P., AND JOHNSON, R. Games without frontiers: Investigating video games as a covert channel. In *2016 IEEE European Symposium on Security and Privacy* (Saarbrucken, Germany, 2016), IEEE, pp. 63–77.

[22] HAYES, J., AND DANEZIS, G. k-fingerprinting: A robust scalable website fingerprinting technique. In *Proceedings of the 25th USENIX Security Symposium* (Austin, TX, USA, 2016), pp. 1187–1203.

[23] HOUMANSADR, A., BRUBAKER, C., AND SHMATIKOV, V. The parrot is dead: Observing unobservable network communications. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy* (San Francisco, CA, USA, 2013), pp. 65–79.

[24] HOUMANSADR, A., NGUYEN, G. T., CAESAR, M., AND BORISOV, N. Cirripede: Circumvention infrastructure using router redirection with plausible deniability. In *Proceedings of the 18th ACM Conference on Computer and Communications Security* (Chicago, IL, USA, 2011), pp. 187–200.

[25] HOUMANSADR, A., RIEDL, T. J., BORISOV, N., AND SINGER, A. C. I want my voice to be heard: IP over Voice-over-IP for unobservable censorship circumvention. In *Proceedings of the 20th Annual Network & Distributed System Security Symposium* (San Diego, CA, USA, 2013).

[26] KAREV, D., MCCUBBIN, C., AND VAULIN, R. Cyber threat hunting through the use of an isolation forest. In *Proceedings of the 18th International Conference on Computer Systems and Technologies* (Ruse, Bulgaria, 2017), pp. 163–170.

[27] KARLIN, J., ELLARD, D., JACKSON, A., JONES, C., LAUER, G., MANKINS, D., AND STRAYER, T. Decoy routing: Toward unblockable Internet communication. In *Proceedings of the USENIX Workshop on Free and Open Communications on the Internet* (San Francisco, CA, USA, 2011).

[28] KULLBACK, S., AND LEIBLER, R. A. On information and sufficiency. *The Annals of Mathematical Statistics 22*, 1 (1951), 79–86.

[29] LI, S., SCHLIEP, M., AND HOPPER, N. Facet prototype. https://magicle.github.io/facet/index.html, 2014. Last Accessed: 2018-02-05.

[30] LI, S., SCHLIEP, M., AND HOPPER, N. Facet: Streaming over videoconferencing for censorship circumvention. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society* (Scottsdale, AZ, USA, 2014), pp. 163–172.

[31] LIU, F. T., TING, K. M., AND ZHOU, Z.-H. Isolation forest. In *Proceedings of the 8th IEEE International Conference on Data Mining* (Pisa, Italy, 2008), pp. 413–422.

[32] MARKOU, M., AND SINGH, S. Novelty detection: a review—part 2:: neural network based approaches. *Signal processing 83*, 12 (2003), 2499–2521.

[33] MCPHERSON, R., HOUMANSADR, A., AND SHMATIKOV, V. CovertCast prototype. https://www.cs.cornell.edu/~shmat/covertcast/, 2016. Last Accessed: 2018-02-05.

[34] MCPHERSON, R., HOUMANSADR, A., AND SHMATIKOV, V. CovertCast: Using live streaming to evade internet censorship. In *Proceedings on Privacy Enhancing Technologies* (Darmstadt, Germany, 2016), vol. 2016(3), pp. 212–225.

[35] MICHALSKI, R., CARBONELL, J., AND MITCHELL, T. *Machine Learning: An Artificial Intelligence Approach*. No. vol. 1. Elsevier Science, 2014.

[36] MOGHADDAM, H., LI, B., DERAKHSHANI, M., AND GOLDBERG, I. Skypemorph: Protocol obfuscation for Tor bridges. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security* (Raleigh, NC, USA, 2012), pp. 97–108.

[37] MOLNÁR, S., AND PERÉNYI, M. On the identification and analysis of skype traffic. *International Journal of Communication Systems 24*, 1 (2011), 94–117.

[38] NGUYEN, T. T., AND ARMITAGE, G. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys & Tutorials 10*, 4 (2008), 56–76.

[39] PANCHENKO, A., AND LANZE, F. Website fingerprinting at internet scale. In *Proceedings of the 23rd Annual Network and Distributed System Security Symposium* (San Diego, CA, USA, 2016).

[40] PEARSON, K. X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 50*, 302 (1900), 157–175.

[41] QUINLAN, J. R. Induction of decision trees. *Machine Learning 1*, 1 (1986), 81–106.

[42] RIMMER, V., PREUVENEERS, D., JUAREZ, M., VAN GOETHEM, T., AND JOOSEN, W. Automated website fingerprinting through deep learning. In *Proceedings of the 25th Annual Network & Distributed System Security Symposium* (San Diego, CA, USA, 2018).

[43] RUBNER, Y., TOMASI, C., AND GUIBAS, L. J. The Earth Mover's Distance As a Metric for Image Retrieval. *International Journal of Computer Vision 40*, 2 (Nov. 2000), 99–121.

[44] SANDULESCU, V., AND CHIRU, M. Predicting the future relevance of research institutions-the winning solution of the kdd cup 2016. *arXiv preprint arXiv:1609.02728* (2016).

[45] SCHÖLKOPF, B., PLATT, J. C., SHAWE-TAYLOR, J. C., SMOLA, A. J., AND WILLIAMSON, R. C. Estimating the support of a high-dimensional distribution. *Neural Computation 13*, 7 (July 2001), 1443–1471.

[46] SCHUSTER, R., SHMATIKOV, V., AND TROMER, E. Beauty and the burst: Remote identification of encrypted video streams. In *Proceedings of the 26th USENIX Security Symposium* (Vancouver, BC, Canada, 2017).

[47] WALT, S. V. D., COLBERT, S. C., AND VAROQUAUX, G. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering 13*, 2 (2011), 22–30.

[48] WANG, L., DYER, K. P., AKELLA, A., RISTENPART, T., AND SHRIMPTON, T. Seeing through network-protocol obfuscation. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (Denver, CO, USA, 2015), pp. 57–69.

[49] WANG, T., CAI, X., NITHYANAND, R., AND JOHNSON, R. Effective attacks and provable defenses for website fingerprinting. In *Proceedings of the 23rd USENIX Security Symposium* (San Diego, CA, USA, 2014), pp. 143–157.

[50] WINTER, P., HERMANN, E., AND ZEILINGER, M. Inductive intrusion detection in flow-based network data using one-class support vector machines. In *Proceedings of the IEEE 4th IFIP International Conference on New Technologies, Mobility and Security* (Paris, France, 2011), pp. 1–5.

[51] WRIGHT, C. V., BALLARD, L., MONROSE, F., AND MASSON, G. M. Language identification of encrypted VoIP traffic: Alejandra y Roberto or Alice and Bob? In *Proceedings of 16th USENIX Security Symposium* (Boston, MA, 2007), pp. 4:1–4:12.

[52] WRIGHT, C. V., COULL, S. E., AND MONROSE, F. Traffic morphing: An efficient defense against statistical traffic analysis. In *Proceedings of the 16th Network and Distributed Security Symposium* (San Diego, CA, USA, 2009), pp. 237–250.

[53] WUSTROW, E., SWANSON, C. M., AND HALDERMAN, J. A. Tapdance: End-to-middle anticensorship without flow blocking. In *Proceedings of the 23rd USENIX Security Symposium* (San Diego, CA, 2014), pp. 159–174.

[54] WUSTROW, E., WOLCHOK, S., GOLDBERG, I., AND HALDERMAN, J. A. Telex: Anticensorship in the network infrastructure. In *Proceedings of the 20th USENIX Security Symposium* (San Francisco, CA, USA, 2011).

[55] YISROEL MIRSKY, TOMER DOITSHMAN, Y. E., AND SHABTAI, A. Kitsune: An ensemble of autoencoders for online network intrusion detection. In *Proceedings of the 25th Annual Network & Distributed System Security Symposium* (San Diego, CA, USA, 2018).

[56] ZHAO, M., AND SALIGRAMA, V. Anomaly detection with score functions based on nearest neighbor graphs. In *Advances in Neural Information Processing Systems* (Vancouver, B.C., Canada, 2009), pp. 2250–2258.

[57] ZHOU, W., HOUMANSADR, A., CAESAR, M., AND BORISOV, N. Sweet: Serving the web by exploiting email tunnels. In *Proceedings of the 6th Workshop on Hot Topics in Privacy Enhancing Technologies* (Bloomington, IN, USA, 2013).

A Appendix

Listing 1 indexes the feature set obtained from the calculation of aggregated statistics from our traffic samples.

```
Summary statistics:
1 - Total number of packets.
2 - Total number of packets - ingress.
3 - Total number of packets - egress.
4 - Total bytes transmitted.
5 - Total bytes transmitted - ingress.
6 - Total bytes transmitted - egress.

Global statistics:
7 - Mean of packet sizes.
8 - Std. deviation of packet sizes.
9 - Variance of packet sizes.
10 - Kurtosis of packet sizes.
11 - Skew of packet sizes.
12 - Maximum packet size.
13 - Minimum packet size.
14:22 - (10-90) percentile of packet sizes.

23 - Mean of packet times.
24 - Std. deviation of packet times.
25 - Variance of packet times.
26 - Kurtosis of packet times.
27 - Skew of packet times.
28 - Maximum packet times.
29 - Minimum packet times.
30:38 - (10-90) percentile of packet times.

Statistics for ingress/egress traffic:
39:70 - 7:38 computed over ingress traffic only.
71:102- 7:38 computed over egress traffic only.

Ingress Packet bursts statistics:
103 - Total number of bursts.
104 - Mean burst size.
105 - Std. deviation of burst sizes.
106 - Variance of burst sizes.
107 - Maximum burst size.
108 - Kurtosis of burst sizes.
109 - Skew of burst sizes.
110:118 - (10-90) percentile of burst sizes.

Ingress Bytes bursts statistics:
119 - Mean bytes transmitted across bursts.
120 - Std. deviation of bytes transmitted across bursts.
121 - Variation of bytes transmitted across bursts.
122 - Kurtosis of bytes transmitted across bursts.
123 - Skew of bytes transmitted across bursts.
124 - Maximum number of bytes in a burst.
125 - Minimum number of bytes in a burst.
126:134 - (10-90) percentile of bytes transmitted.

Egress Packet bursts statistics:
135 - Total number of bursts.
136 - Mean burst size.
137 - Std. deviation of burst sizes.
138 - Variance of burst sizes.
139 - Maximum burst size.
140 - Kurtosis of burst sizes.
141 - Skew of burst sizes.
142:150 - (10-90) percentile of burst sizes.

Egress Bytes bursts statistics:
151 - Mean bytes transmitted across bursts.
152 - Std. deviation of bytes transmitted across bursts.
153 - Variation of bytes transmitted across bursts.
154 - Kurtosis of bytes transmitted across bursts.
155 - Skew of bytes transmitted across bursts.
156 - Maximum number of bytes in a burst.
157 - Minimum number of bytes in a burst.
158:166 - (10-90) percentile of bytes transmitted.
```

Listing 1: Summary statistics considered as features.