Mobile Traffic Anonymization Through Probabilistic Distribution

Conference Paper · February 2019

DOI: 10.1109/ICIN.2019.8685871

4 authors, including:

Louma Chaddad
American University of Beirut
4 PUBLICATIONS **2** CITATIONS
SEE PROFILE

Ali Chehab
American University of Beirut
285 PUBLICATIONS **1,468** CITATIONS
SEE PROFILE

Imad H. Elhajj
American University of Beirut
177 PUBLICATIONS **1,788** CITATIONS
SEE PROFILE

Mobile Traffic Anonymization Through Probabilistic Distribution

Louma Chaddad, Ali Chehab, Imad H. Elhajj, and Ayman Kayssi
Department of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon
Correspondance : {lac07, chehab, ie05, ayman}@aub.edu.lb

***Abstract*—Current implementations of mobile apps offer limited security assurances against traffic analysis. Encryption is not effective in hiding particular patterns within packets that can be used as side-channel information to classify specific apps. What we need is an anonymity system that ensures strong security with acceptable computational overhead and latency for interactive app usage. Recently, we undertook this problem by mutating an app traffic that we try to defend so that it resembles the traffic of another app. Our goal in this paper is to develop a simpler and more scalable system to anonymize mobile app packet traffic without the need of another app's model traffic. This could happen using probabilistic distribution of packet sizes. We propose first a scheme that regenerates statistical modeling of app packet lengths. Then, we present a privacy preserving technique that implements defense against traffic analysis and confines the incurred overhead by mutation of packet lengths of the incoming traffic to the regenerated ones. Experiments show that using this confusion technique, we are able to reduce 91.1% classification accuracy to 0.9% with only 12.51% overhead. On the other hand, we demonstrate a second technique for anonymization where we thwart traffic analysis of mobile app traffic by modifying its packet sizes probability distribution to another dissimilar distribution.**

Keywords— Side-channel information; obfuscation; anonymization; traffic padding; traffic classification; packet length statistics; probability distribution function; modeling.

I. INTRODUCTION

Mobile apps have affected multiple aspects of our lives. Whether it is about shopping, ordering food, hiring cabs, checking bank accounts, communicating with others, etc., mobile phones or what we rightly call "smartphones" are the first devices that we think of to accomplish all these activities. This is because of mobile apps that can be downloaded and deployed easily.

In recent past years, research on network traffic classification has increased [13, 14]. Using statistical characteristics of mobile app traffic, identification of the exact apps installed and used by a victim's phone is now possible. This is because of traffic analysis that allows to accurately identify these apps from the encrypted traffic they generate. Hence, privacy of users using mobile apps is really pressing and becoming an intricate puzzle. In fact, data leakage is a worrisome threat for both mobile users and programmers. The former because it exposes their sensitive information, and the latter due to challenges of ensuring security without degrading user experience.

A common method to defeat this threat is by padding payload traffic to its maximum size. Another strategy consists of inserting dummy packets to original traffic patterns [15]. These methods make statistics of the overall traffic different from that of the original traffic. However, they are unsatisfactory and can be defeated. In addition, they introduce large overhead and are not practical for deployment [12].

Besides, Tor is currently a well-known privacy security solution that uses layered encryption and promises anonymity by routing data through several overlay hops. However, [8]-[11] show that Tor does not offer sufficient security against mobile apps fingerprinting and is vulnerable to traffic analysis.

In this work, we present network-based privacy prevention models for thwarting statistical traffic analysis of mobile devices. Our anonymization techniques obfuscate mobile traffic and decrease accuracy of classifiers that identify traffic classes. The main objective of our proposed system is to preserve privacy of mobile apps so that traffic analysis is confused. We develop a new way for mutation, by controlling the probability of packet lengths. This is equivalent to imposing probabilistic end-to-end lengths in a typical network. Specifically, we make the following contributions:

- We identify probability distribution models of app packet lengths that fits it correctly.
- We regenerate an app packet sizes from its best-fitted distribution model.
- In our first anonymization technique, we mutate packet lengths of a source app that we are trying to defend to generated packet sizes of another target app.
- In our second obfuscation technique, we change probability distribution of packet sizes to a different one without the need of another target app.
- We validate the efficiency of our approaches on real app traffic.
- We assess the efficiency of our evasion technique by comparing it to our previous model in [18].

The rest of this paper is organized as follows. Section II describes prior work on obfuscation techniques. Section III describes our attack model, and then explains our solution approaches. In section IV, we present our probability distribution profiling experiment of mobile apps packet lengths. Section V validates via experiments our thwarting algorithms. Section VI compares our evasion model against our recently

978-1-5386-8336-1/19/$31.00 ©2019 IEEE

published thwarting technique. Finally, Section VII concludes our work.

II. Literature Review

The study of packet length distribution is a fundamental step for traffic modeling [20]. A large body of work in the literature studies network traffic properties. In [19], authors report packet size distribution for network flows at the Internet layer, transport layer, and application layer. According to [3], Pareto Second Kind distribution is best fit for network packet inter-arrival time distribution. In [2], the authors find that lognormal and GEV models are the optimal statistical models characterizing Internet traffic. According to [4], network traffic model for non-congested Internet backbone links can be described as a Poisson short-noise process. In [7], the authors demonstrate that models can be used to approximate distribution of network traffic. They suggest a probability density function model to fit packet lengths in computer networks. In addition, they confirm that Exponential, Lognormal, Pareto of Weibull distributions can be used for the same purpose.

These clues to a conclusion that there are different models with different characteristics that can capture network traffic. Hence, there is not a single model that can be used. Standard goodness-of-fit tests such as Kolmogorov-Smirnov, Anderson-Darling, and Chi-Square [5] allow a mathematical proof for the optimal fit.

On the other hand, identifying network traffic has gained much attention in the last few years. Work in [6] used packet size distribution to classify TCP application. In [21], the authors depend on lengths of the first packets of SSL or TLS sessions to discriminate between network applications.

In addition, the literature presents many classification techniques that identify encrypted traffic using machine learning. These traffic analysis methods build classifiers based on metadata such as timing, traffic patterns, traffic direction, and volume. The majority of these techniques rely on packet length feature [22]-[25]. In fact, packet length reveals characteristics of the underlying class and as such is useful in discriminating between different types of network traffic.

The issue of identifying traffic classes is sensitive from security aspect especially in the field of mobile devices. This is because identifying apps on a mobile phone can reveal a lot of information about the user's habits, preferences, activities, religious beliefs, or medical conditions and so on. Most techniques in the literature to defend against it focus on computer and web browsing applications.

For instance, [26] suggests using virtualized MAC addresses to obscure an adversary's analysis of traffic through Wi-Fi connection on a computer. To prevent this attack on web browsing in [27], the authors add a random amount of padding to the encrypted packets. However, [28] shows later that this countermeasure does not actually realize an efficient covering of all traffic features. Another strategy consists of injecting dummy packets to modify statistical analysis [29, 30]. However, this method adds overhead that is not acceptable for smartphones use. Wright et al. [16] propose morphing one class of traffic to appear like another class. They realize it using convex optimization techniques. However, this method adds latency and is computationally expensive. In our previous work [18], we confuse an app traffic by mutating packet lengths of each flow so that they look like another app. To realize this, we start by shaping an app's smallest packet length in a flow to the smallest packet length of another app's flow. Successively, we modify lengths of all the packets in the flow. Using this method, a classifier's accuracy of 91.1% is reduced to 15.7% with only 34.6% overhead.

III. Traffic Analysis Attack and Our Anonymization Solution

We consider a traffic-analysis attack scenario where a local adversary collects app traffic data from a victim mobile user. The attacker examines side-channel information (IP packet headers and metadata) from the encrypted mobile app traffic. He relies on classifiers using machine-learning techniques to draw conclusions about the exact app type. The above scenario raises the need to defend the security of a mobile network that is under the scrutiny of a passive attacker.

Modeling app traffic plays an important role in network design and planning. In addition, these models can be used to create synthetic traffic. Authors in [1] show that any Internet traffic has self-similarity property. This implies that different parts of Internet traffic are either exactly the same or similar. Consequently, we use captured sessions of mobile app traffic communication to conduct our experiment.

Initially, we identify a proper model that fits packet lengths of different apps traffic and distributions that do not fit correctly. It is worth noting that there is no single model that can describe efficiently traffic in all types of networks [3]. In fact, there are various number of probability distribution models that capture correct features and characteristics of traffic. We will prove the optimal model mathematically using Kolmogorov-Smirnov standard goodness-of-fit test, which represents the maximum absolute difference between distribution and experimental curves. Lesser KS indicates the better is the fit. Once the proper distribution fit is determined, app packet lengths of mobile app traffic are regenerated accordingly.

Our goal is to protect against statistical traffic analysis by confusing an app traffic. In our first model, we mutate flows generated by the source app to defend so that their distribution becomes similar to regenerated packet lengths of another target app. For each incoming packet from the source app, we compare its size s_l to the size t_l of the regenerated length in the target app. If s_l is less than t_l, we proceed by padding s_l with zeroes such that the resultant packet length is the same as t_l. Otherwise, we fragment the packet into 2 fragments: first fragment has the same size as t_l, second fragment is considered as a new incoming packet where its size is compared to the next generated packet length of the target app.

We have considered app traffic mutation to thwart traffic classification by mutating a source app into a different target app. We could consider app traffic mutation of an app to itself

without the need of a target app by only changing the distribution of its lengths. This would make our model more scalable. It rejects the need for a second app, especially if the type of the app that we are trying to evade is unknown. Therefore, in our second model, we only need to study the distribution of the app's packet lengths and change it to a different one. To realize this, we identify a distribution model that does not fit packet lengths of the app traffic in consideration, and we regenerate packet lengths from it. Lengths of regenerated packets are considered as 'target' app traffic lengths. We apply the mutation method explained previously to mutate a source packet length to the target packet length.

IV. Mobile Apps Probability Distribution Profiling

In our experiments, we use network traffic traces of 6 widely used apps that are collected on Android smartphones and are manually classified [17]. Table 1 summarizes types of the 6 apps as well as the actions executed while capturing their traffic traces.

TABLE 1. EXPERIMENT DATASET

#	App	Type	Actions
1	Skype video call	Video Call	Video Call
2	Facebook browsing	Interactive browsing	Browse, Comment, Like, Post
3	8 ball pool/ Game	Game	Play
4	WhatsApp messaging	Messaging	Texting, media sharing
5	Viber VoIP	VOIP	Voice Call
6	YouTube streaming	Video streaming	Play videos

First, we examine the distribution of packet lengths across all instances in each app traffic set, and we estimate its probability distribution. Graphs in figure 1 show the statistical analysis results of app traffic packet sizes distribution for the 6 apps of our dataset.

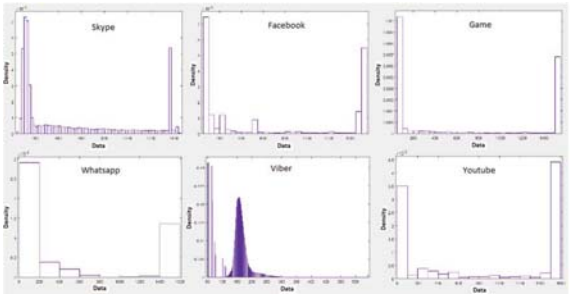

Figure 1. App traffic packet sizes distribution

Next, we examine the best-fit function for each app packet sizes distribution. We model different distributions for the 6 different apps traffic. We consider Normal, Poisson, Rician, and Weibull as they are used in computer network traffic modeling. Normal is described by location and scale parameters. Poisson uses mean parameter. Rician is characterized by non-centrality and scale parameters. Weibull uses shape parameter and scale parameter.

We model these distributions for packet lengths of Skype and Viber in Figure 2 and 3, respectively. We also specify the parameters used in Table 2. Param 1 and Param 2 represent the first and second parameter of distribution, respectively. Each class among Normal, Poisson, Rician, and Weibull hardly fit the data distribution perfectly. However, by visual examination of the fitted curves displayed in Figure 2, Normal is perceived to be the best-fit distribution over Skype traffic. Viber is observed to be closer to Poisson distribution in Figure 3. By simply plotting the 4 distributions fits for the remaining 4 apps, we notice that neither distribution matches.

Next, we evaluate, using goodness-of-fit function, which distribution-fit accurately describes traffic in our traces. At this purpose, we use Kolmogorov-Smirnov test to prove that a Normal process accurately describes traffic in Skype, and that a Poisson process accurately identifies Viber traffic. In fact, Normal showed the lowest KS parameter in case of Skype, and Poisson has the lowest KS parameter in case of Viber as presented in Table 2.

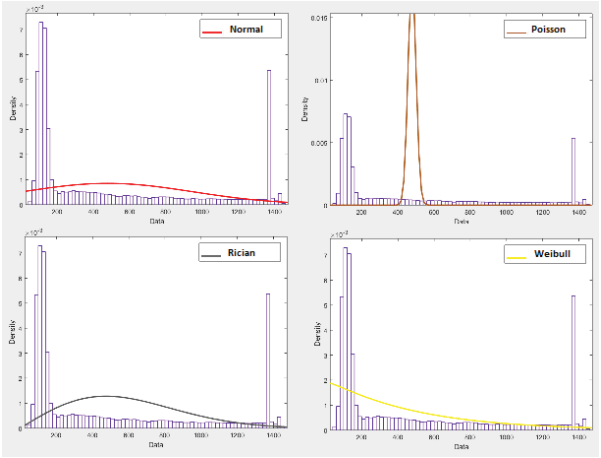

Figure 2. Distribution fits with respect to Skype packet lengths

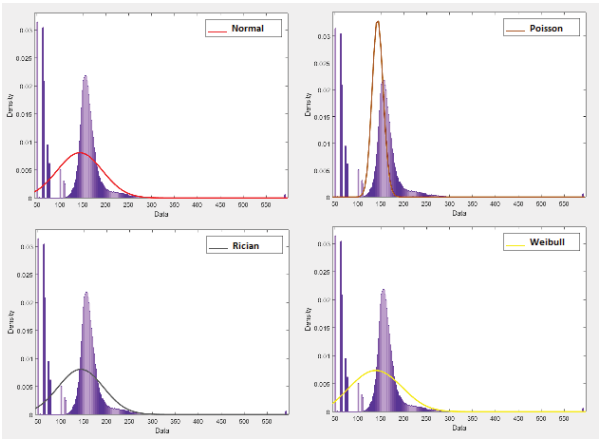

Figure 3. Distribution fits with respect to Viber packet lengths

Table 2 Packet lengths Distribution Fit Parameters and Goodness-of-fit

App	Distribution	Param 1	Param 2	KS
Skype	Normal	Location= 478.3	Scale= 469.9	0.217
	Poisson	Mean= 478.3	-	0.944
	Rician	Noncentrality= 27.8	Scale=473.8	0.375
	Weibull	Shape=483.4	Scale=1.0	0.329
Viber	Normal	Location= 143.3	Scale= 49.3	0.485
	Poisson	Mean= 143.3	-	0.116
	Rician	Noncentrality= 132.5	Scale= 52.0	0.526
	Weibull	Shape= 159.2	Scale= 2.9	0.583

V. Obfuscation Experiments and Results

To evaluate our evasion defense algorithms, we use the app classification experiment discussed in our previous work [18]. We create 4 classifier models to classify app traffic types. The accuracy of the different classifiers and their corresponding parameters are shown in Table 3.

TABLE 3. DATASET ACCURACY

Model	Parameters	Accuracy
SVM	Quadratic Kernel	74.7%
Bagged Trees	Min Leaf Size =2	90.0%
KNN	# of neighbors=5; Distance Weight=Squared inverse	83.9%
Random Forest	Min Leaf Size =2	91.1%

Next, we describe two experiments corresponding to two different obfuscation techniques. The difference between the 2 models is the target app used in mutation.

In our first experiment, we calculate the mean and standard deviation of real Skype packet lengths to regenerate them from normal distribution. We also calculate the mean of Viber packet lengths to regenerate them from Poisson distribution. The regeneration of Skype packet lengths from Normal distribution comes at the cost of 1.6% overhead only in comparison with the original Skype traffic. Also, regenerating Viber packet sizes from Poisson distributions costs 0.259% only when we compare it to the original Viber traffic.

Packets of our dataset are transferred one after the other and several flow sessions of these packets occur at the same time. We sort packets for each flow session between two communicating entities. Applying our first obfuscation technique, we mutate, for each flow session, the source app traffic to regenerate packet lengths from Normal distribution of Skype. We pad with zeroes each incoming packet from the source app to a generated packet length in case it is smaller. Otherwise, we proceed by fragmenting it, as discussed in section III. We test the 6 apps consecutively as source apps to mutate. Using our 4 model classifiers, we predict the classes of the modified traffic. Results of prediction for Quadratic SVM, Bagged Tree, Fine KNN, and Random Forest are shown in Tables 4, 5, 6, and 7, respectively for the six different apps. We present the resulting overhead from mutating each source app packet lengths to Normal distribution of Skype packet sizes in Table 8.

Simulation results show for example that mutating Game packet lengths to Normal distribution of Skype packet sizes can decrease SVM classifier's accuracy from 76.7% to 0.4% as shown in Table 4, Bagged Trees classifier's accuracy from 90% to 0.7% as shown in Table 5, KNN classifier's accuracy from 83.9% to 2.76% as shown in Table 6, and Random Forest classifier's accuracy from 91.1% to 0.9% as shown in Table 7 with only 12.51% average overhead as presented in Table 8. Hence, drastic drop of the accuracy is achieved with a minimal cost of traffic volume increase, by a factor 1.12. Also, 80.96% of the Game traffic was confused by Bagged Tree as Skype traffic as shown in Table 5. This proves the efficiency of our algorithm to morph a Game traffic class into Skype class.

Table 4. Confusion of Source app mutated to Normal distribution of Skype packet sizes by Quadratic SVM

Predicted as / Source app	Skype	FB	Game	Whtsapp	Viber	Youtube
Skype	35.47%	43.59%	0.47%	3.09%	7.5%	9.88%
Facebook	49.42%	39.49%	0.15%	1.84%	2.86%	6.25%
Game	43.22	34.25%	0.4%	4.37%	6.67%	11.09%
Whatsapp	32.84	28%	0.42%	1.89%	11.37%	25.47%
Viber	42.07%	40.71%	0.74%	3.32%	3.57%	9.59%
Youtube	35.47%	43.59%	0.47%	3.09%	7.5%	9.88%

Table 5. Confusion of Source app mutated to Normal distribution of Skype packet sizes by Bagged Tree

Predicted as / Source app	Skype	FB	Game	Whtsapp	Viber	Youtube
Skype	55.71%	30.77%	0%	6.35%	0.19%	6.92%
Facebook	16.84%	68.32%	0.35%	6.37%	1.33%	6.8%
Game	80.96%	10.56%	0.7%	3.18%	0.48%	4.12%
Whatsapp	47.13%	24.6%	0%	15.17%	0.69%	12.41%
Viber	32.84	24 %	0%	2.32%	10.53%	30.32%
Youtube	62.36%	15.25%	0.74%	4.31%	1.23%	16.11%

Table 6. Confusion of Source app mutated to Normal distribution of Skype packet sizes by Fine KNN

Predicted as / Source app	Skype	FB	Game	Whtsapp	Viber	Youtube
Skype	47.89%	22.23%	1.8%	8.09%	3.75%	16.255
Facebook	55.72%	23.45%	2.13%	3.05%	2.03%	13.61%
Game	40.46%	23.68%	2.76%	14.25%	2.53%	16.32%
Whatsapp	35.16%	25.26%	1.05%	13.68%	9.68%	15.16%
Viber	51.41%	20.3%	3.32%	5.9%	1.35%	17.71%
Youtube	47.89%	22.23%	1.8%	8.09%	3.75%	16.255

Table 7. Confusion of Source app mutated to Normal distribution of Skype packet sizes by Random Forest

Predicted to be / Source app	Skype	FB	Game	Whtsapp	Viber	Youtube
Skype	57.23%	28.21%	0%	5.73%	0.15%	8.64%
Facebook	22.24%	58.91%	0.71%	5.98%	2.13%	9.8%
Game	76.12%	9.23%	0.9%	5.90%	0.58%	7.19%
Whatsapp	49.2%	28.69%	0%	17.78%	0.71%	3.41%
Viber	34.51%	36 %	0%	1.12%	10.11%	18.24%
Youtube	59.9%	17.79%	0.81%	5.12%	1.76%	13.51%

Table 8. Overhead resulting from mutating to Normal distribution of Skype packet sizes

Source app / Target	Skype	Facebook	Game	WhatsApp	Viber	YouTube
Normal distribution of Skype	1.6 %	5.28 %	12.51%	35.26%	63.34%	13.37%

Similarly, we mutate in each session the source app traffic to regenerated packet lengths from Poisson distribution of Viber. Using our 4 model classifiers, we predict the classes of the modified traffic. Results of prediction for Quadratic SVM, Bagged Tree, Fine KNN, and Random Forest are shown in Tables 9, 10, 11, and 12, respectively for the six different apps. We present the resulting overhead from mutating each source app packet lengths to Poisson distribution of Viber packet sizes in Table 13.

Table 9. Confusion of Source app mutated to Poisson distribution of Viber packet sizes by Quadratic SVM

Predicted as / Source app	Skype	FB	Game	Whtspp	Viber	Youtube
Skype	30.00%	42.31%	0%	1.35%	24.81%	1.54%
Facebook	36.54%	38.65%	0%	1.54%	21.73%	1.54%
Game	9.11%	64.24%	2.66%	0.15%	23.3%	0.53%
Whatsapp	22.99%	66.67%	0.46%	2.3%	7.13%	0.46%
Viber	6.53%	9.05%	0%	0.21%	84.21%	0%
Youtube	6.89%	64.21%	0.25%	0.62%	27.18%	0.86%

Table 10. Confusion of Source app mutated to Poisson distribution of Viber packet sizes by Bagged Tree

Predicted as / Source app	Skype	FB	Game	Whtspp	Viber	Youtube
Skype	52.69%	22.61%	0%	5.38%	18.85	0.38%
Facebook	17.81%	68.2%	0%	7.15%	6.8%	0%
Game	7.56%	83.72%	0.48%	7.27%	0.97%	0%
Whatsapp	24.6%	53.79%	0%	18.39%	3.22%	0%
Viber	1.68%	6.58%	0%	3.16%	88.58%	0%
Youtube	10.95%	71.96%	0%	12.67%	4.18%	0.25%

Table 11. Confusion of Source app mutated to Poisson distribution of Viber packet sizes by Fine KNN

Predicted as / Source app	Skype	FB	Game	Whtspp	Viber	Youtube
Skype	45.00%	31.15%	1.92%	2.12%	19.42%	0.38%
Facebook	21.13%	49.49%	11.56 %	5.04%	11.95%	0.82%
Game	20.45%	20.54%	48.89%	2.76%	4.75%	2.62%
Whatsapp	25.98%	40.46%	7.82%	20.69%	3.45%	1.61%
Viber	1.47%	7.79%	0.21%	2.32%	88.21%	0%
Youtube	13.65%	53.38%	10.95%	8.73%	3.94%	13.65%

Table 12. Confusion of Source app mutated to Poisson distribution of Viber packet sizes by Random Forest

Predicted as / Source app	Skype	FB	Game	Whtsapp	Viber	Youtube
Skype	53.48%	24.31%	0%	4.21%	17.54%	0.38%
Facebook	19.84%	65.3%	0%	6.54%	6.41%	1.01%
Game	5.09%	84.92%	1.43%	8.05%	0.71%	0%
Whatsapp	21.6%	55.39%	0%	16.99%	4.51%	0%
Viber	1.58%	7.32%	0%	4.88%	85.87%	0%
Youtube	9.06%	72.57%	0%	13.28%	5.09%	0.14%

Table 13. Overhead resulting from mutating to Poisson distribution of Viber packet sizes

Source app / Target	Skype	Facebook	Game	WhatsApp	Viber	YouTube
Poisson distribution of Viber	19.87%	7.72%	14.33%	9.87%	0.0259%	3.81%

Simulation results show for example that mutating Game packet lengths to Poisson distribution of Skype packet sizes can decrease SVM classifier's accuracy from 76.7% to 2.66% as shown in Table 9, Bagged Trees classifier's accuracy from 90% to 0.48% as shown in Table 10, KNN classifier's accuracy from 83.9% to 48.89% as shown in Table 11, and Random Forest classifier's accuracy from 91.1% to 1.43% as shown in Table 12 with only 14.33% average overhead as presented in Table 8. Hence, drastic drop of the accuracy is achieved with a minimal cost of traffic volume increase, by a factor 1.14.

In our second experiment, we evaluate another obfuscation technique where we change the distribution of packet lengths of an app into a dissimilar one. We discussed earlier in section IV that apps other than Skype or Viber do not fit correctly into Normal or Poisson distributions. Therefore, we mutate packet lengths of Facebook, Game, WhatsApp, and YouTube to regenerated Normal or Poisson distributions of their packet sizes. We concentrate on Normal distribution and Poisson distribution for simple modeling. Prediction results of classes of the modified app traffic of Facebook, Game, WhatsApp, and Game using our 4 classifiers are presented in Tables 14, 15, 16, and 17, respectively.

This obfuscation technique results in less but acceptable attack evasion outcomes compared to our first one presented earlier. For example, when predicting Game class after mutating its packet sizes to Normal distribution, SVM classifier's accuracy decreases from 76.7% to 0.68%, Bagged Trees classifier's accuracy from 90% to 0.19%, KNN classifier's accuracy from 83.9% to 4.55%, and Random Forest classifier's accuracy from 91.1% to 0.28% with only 7.73% average overhead. Also, when predicting Game class after mutating its packet sizes to Poisson distribution, SVM classifier's accuracy decreases from 76.7% to 2.23%, Bagged Trees classifier's accuracy from 90% to 1.7%, KNN classifier's accuracy from 83.9% to 12.16%, and Random Forest classifier's accuracy from 91.1% to 1.9% with only 29.1% average overhead.

Table 14. Prediction of mutated Facebook packet lengths to itself using Normal or Poisson distributions

Algorithm	SVM	BT	KNN	RF	Ovrhd
Original Facebook accuracy	76.7%	90%	88.9%	91.1%	-
Facebook **Mutated** to Facebook with lengths generated from Normal distribution	46.56%	68.48%	46.91%	65.2%	7.33%
Facebook **Mutated** to Facebook with lengths generated from Poisson distribution	52.93%	59.65%	50.27%	58.3%	29.90%

Table 15. Prediction of mutated Game packet lengths to itself using Normal or Poisson distributions

Algorithm	SVM	BT	KNN	RF	Ovrhd
Original Game accuracy	76.7%	90%	88.9%	91.1%	-
Game **Mutated** to Game with lengths generated from Normal distribution	0.68%	0.19%	4.55%	0.28%	7.73%
Game **Mutated** to Game with lengths generated from Poisson distribution	2.23%	1.7%	12.16%	1.9%	29.10%

Table 16. Prediction of mutated WhatsApp packet lengths to itself using Normal or Poisson distributions

Algorithm	SVM	BT	KNN	RF	Ovrhd
Original WhatsApp accuracy	76.7%	90%	88.9%	91.1%	-
WhatsApp **Mutated** to WhatsApp with lengths generated from Normal	11.72%	15.63%	18.62%	16.11%	32.62%
WhatsApp **Mutated** to WhatsApp with lengths generated from Poisson	3.68%	15.86%	30.34%	0.97%	21.75

Table 17. Prediction of mutated YouTube packet lengths to itself using Normal or Poisson distributions

Algorithm	SVM	BT	KNN	RF	Ovrhd
Original YouTube accuracy	76.7%	90%	88.9%	91.1%	-
YouTube **Mutated** to YouTube with lengths generated from Normal	37.88%	13.78%	9.59%	12.86%	6.77%
YouTube **Mutated** to YouTube with lengths generated from Poisson	1.11%	4.18%	5.04%	4.89%	4.52%

VI. App Traffic Anonymization Assessment

In this section, we assess the effectiveness of our evasion models by comparing them to MTU quantization, and to the mutation algorithm discussed in our previous work [18].

We simulate MTU quantization by padding app packets sizes to their maximum transmit unit (MTU). Also, we simulate our model in [18] where we choose Skype as the source app and Game as the target app. We predict using our 4 classifiers, described in section V, the classes of mutated app traffic.

As presented in Table 18, The MTU quantization scheme decreases SVM classifier's accuracy from 76.7% to 32.9%, Bagged Trees classifier's accuracy from 90 to 21.3%, KNN's classifier from 83.9% to 26.2%, and Random Forest classifier's accuracy from 91.1% to 22.3% with an overhead of 216.6%. This large overhead would lead to degradation of performance of the network.

Our previous work in [18], decreases SVM classifier's accuracy from 76.7% to 14.2%, Bagged Trees classifier's accuracy from 90 to 19.4%, KNN's classifier from 83.9% to 22.1%, and Random Forest classifier's accuracy from 91.1% to 15.7% with an overhead of 34.6%.

Our algorithm results, presented in section V, reduce the classifiers' accuracy drastically with an acceptable overhead. Hence, countermeasures presented in this work are more efficient and suitable for mobile devices use, especially that these devices cannot handle excessive overhead.

Table 18. Results of previous evasion models

Alg	SVM	BT	KNN	RF	Ovrhd
Original % Acc	76.7	90.0	83.9	91.1	-
% Acc of MTU	32.9	21.3	26.2	22.3	216.6%
% Acc of Alg in [18]	14.2	19.4	22.1	15.7	34.6%

VII. Conclusion

In this paper, we analyze apps traffic in terms of packet lengths distribution. We present a statistical modeling for packet lengths of mobile app traffic. Statistical tests over real mobile app traffic demonstrate that Normal and Poisson distributions characterize accurately packet lengths distributions of Skype and Viber, respectively.

We propose a novel approach relying on traffic statistics to mutate packet traffic lengths to defend against traffic analysis attacks, while satisfying quality of service requirements. Experiment evaluation shows that 91.1 % accuracy of a classifier is reduced to 0.9% using our thwarting technique with 12.51% overhead only. Our approach clearly outperforms the other state-of the art defense solutions against traffic analysis.

References

[1] G. He, and J. C. Hou, "On sampling self-similar Internet traffic," Computer Networks, 50.16 (2006), pp. 2919-2936.

[2] M. Alasmar, and N. Zakhleniuk, "Network Link Dimensioning based on Statistical Analysis and Modeling of Real Internet Traffic," arXiv preprint arXiv:1710.00420 , 2017.

[3] Garsva, E., et al. "Packet inter-arrival time distribution in academic computer network," Elektronika ir Elektrotechnika 20.3 (2014), pp. 87-90.

[4] C. Barakat, P. Thiran, G. Iannaccone, C. Diot, P. Owezarski, "Modeling internet backbone traffic at the flow level," IEEE Trans. Signal Processing, vol. 51, no. 8, pp. 2111–2124, Aug. 2003.

[5] D'Agostino, RalphB. Goodness-of-fit-techniques. Routledge, 2017.

[6] E. Rocha, P. Salvador, and A. Nogueira, "Detection of Illicit Network Activities based on Multivariate Gaussian Fitting of Multi-Scale Traffic Characteristics," IEEE ICC 2011, 1-6, 2011.

[7] E. Castro, M. Alencar, and I. Fonseca, "Probability density functions of the packet length for computer networks with bimodal traffic," in International Journal of Computer Networks & Communications 5.3: 17, 2013.

[8] K. Kohls, and C. Pöpper, "POSTER: Traffic Analysis Attacks in Anonymity Networks," ACM Asia Conference on Computer and Communications Security, 2017.

[9] F. Mercaldo, and F. Martinelli, "Tor traffic analysis and identification." AEIT International Annual Conference, 2017.

[10] A. Cuzzocrea, et al. "Tor traffic analysis and detection via machine learning techniques," IEEE International Conference on Big Data, 2017.

[11] J. Lingyu, et al. "A hierarchical classification approach for tor anonymous traffic." IEEE 9th International Conference on Communication Software and Networks, (ICCSN), 2017.

[12] M. Nia, and A. Ruiz-Martínez, "Systematic literature review on the state of the art and future research work in anonymous communications systems," Computers & Electrical Engineering (2017).

[13] P. Velan, et al. "A survey of methods for encrypted traffic classification and analysis," International Journal of Network Management 25.5 (2015): 355-374.

[14] M. Conti, et al. "The dark side (-channel) of mobile devices: A survey on network traffic analysis," IEEE Communications Surveys & Tutorials (2018).

[15] Z. Zhuo, et al. "Website Fingerprinting Attack on Anonymity Networks Based on Profile Hidden Markov Model," IEEE Transactions on Information Forensics and Security 13.5, 1081-1095, 2018.

[16] C.V. Wright, S.E. Coull, and F. Monrose, “Traffic morphing: an efficient defense against statistical traffic analysis,” Proceedings of NDSS, pp. 1–14, 2009.

[17] G. Ajaeiya, I. Elhajj, A Chehab, A. Kayssi, and M. Kneppers, "Mobile Apps identification based on network flows," in Knowledge and Information Systems, 1-26, 2017.

[18] L. Chaddad, A. Chehab, I. H. Elhajj, and A. Kayssi, "App traffic mutation: Toward defending against mobile statistical traffic analysis," IEEE INFOCOM on Computer Communications Workshops (INFOCOM WKSHPS) , 2018.

[19] G. Eimantas, N. Paulauskas, and G. Grazulevicius, "Packet size distribution tendencies in computer network flows," IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream), 2015.

[20] F. Tanjeem, M. Sarwar Uddin, and A. Rahman, "Wireless media access depending on packet size distribution over error-prone channels," IEEE International Conference on Networking Systems and Security (NSysS), 2015.

[21] S. Meng, et al. "Classification of encrypted traffic with second-order Markov chains and application attribute bigrams," IEEE Transactions on Information Forensics and Security 12.8, 1830-1843, 2017.

[22] A. Blake, and D. McGrew. "Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity," ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017.

[23] V.F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, “Robust smartphone app identification via encrypted network traffic analysis.,” IEEE Transactions on Information Forensics and Security 13.1, 63-78, 2018.

[24] Y. Fu, X. Hui, L. Xinjiang, Y. Jin, and C. Can, “Usage Classification with Encrypted Internet Traffic in Mobile Messaging Apps.,” IEEE Transactions on Mobile Computing 15.11, 2851-2864, 2016.

[25] V.F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, “Appscanner: Automatic fingerprinting of smartphone apps from encrypted network traffic.,” 1st IEEE European Symposium on Security and Privacy, 2016. (To appear).

[26] F. Zhang, W. He, Y. Chen, Z. Li, X. Wang, S. Chen, and X. Liu, “Thwarting wi-fi side-channel analysis through traffic demultiplexing,” IEEE Transactions on Wireless Comm., vol. 13, no. 1, pp. 86–98, Jan 2014.

[27] A. Back, U. Möller, and A. Stiglic, “Traffic analysis attacks and tradeoffs in anonymity providing systems,” Information Hiding, 2001, pp. 245–257.

[28] K.P. Dyer, S.E. Coull, T. Ristenpart, and T. Shrimpton, “Peek-a-boo, i still see you: why efficient traffic analysis countermeasures fail,” in IEEE Symposium on Security and Privacy, 2012, pp. 332–346.

[29] X. Cai, R. Nithyanand, and R. Johnson, “CS-BuFLO: A Congestion Sensitive Website Fingerprinting Defense.,” in Workshop on Privacy in the Electronic Society (WPES), pages 121–130. ACM, 2014.

[30] X. Cai, R. Nithyanand, T. Wang, R. Johnson, and I. Goldberg, “A Systematic Approach to Developing and Evaluating Website Fingerprinting Defenses,” in ACM Conference on Computer and Communications Security (CCS), pages 227– 238. ACM, 2014.