A Realistic Approach for Network Traffic Obfuscation using Adversarial Machine Learning

Alonso Granados[1], Mohammad Sujan Miah[2], Anthony Ortiz[3], and Christopher Kiekintveld[2]

[1] University of Arizona, Tucson AZ 85721, USA
alonsog@email.arizona.edu
[2] University of Texas at El Paso, El Paso TX 79968, USA
msmiah@miners.utep.edu
cdkiekintveld@utep.edu
[3] Microsoft AI for Good Research Lab, Redmond WA 98052, USA
anthony.ortiz@microsoft.com

Abstract. Adversaries are becoming more sophisticated and standard countermeasures such as encryption are no longer enough to prevent traffic analysis from revealing important information about a network. Advanced encryption techniques are intended to mitigate network information exposure, but they remain vulnerable to statistical analysis of traffic features. An adversary can classify different applications and protocols from the observable statistical properties, especially from the meta-data (e.g. packet size, timing, flow directions, etc.). Several approaches are already being developed to protect computer network infrastructure from attacks using traffic analysis, but none of them are fully effective. We investigate solutions based on obfuscating the patterns in network traffic to make it more difficult to accurately use classification to extract information such as protocols or applications in use. A key problem of using obfuscation methods is to determine an appropriate algorithm that introduces minimal changes but preserves the functionality of the protocol. We apply Adversarial Machine Learning techniques to find realistic small perturbations that can improve the security and privacy of a network against traffic analysis. We introduce a novel approach for generating adversarial examples that obtains state-of-the-art performance compared to previous approaches, while considering more realistic constraints on perturbations.

Keywords: Network Data Analysis · Data Obfuscation · Adversarial Machine Learning

1 Introduction

Heterogeneous network structure and the growing complexity of the IT environment introduce new vulnerabilities to computer networks. One evolving tech-

The first author attended The University of Texas at El Paso during this work.

nique is using statistical traffic analysis to perform reconnaissance. Though advanced encryption techniques limit the information available to traffic analysis, encrypted network traffic can still have observable characteristics like packet sizes and inter-arrival times that reveal useful information to attackers that is a potential threat for network security [12, 4]. Sophisticated adversaries possess knowledge about communication types and maintain databases of well-known traffic patterns and protocols such as UDP, TCP, VoIP, ESP, among others. From the raw traffic, an adversary can determine likely features of the of source/destination without needing to decrypt sensitive information. They can also distinguish statistical characteristics used for communication by different protocols. There are currently no perfect methods to prevent traffic analysis completely. One approach to alleviate this issue is based on deceiving attackers by generating and sending false network traffic along with real traffic, but this can lead to costly overhead.

Recently Deep Packet Inspection (DPI) has become a common technique used by the network administrators to match specific byte patterns from a known database and to update it when unknown patterns are found. However, manually maintaining and updating a dynamic database from the immense network flows is a very tedious task. Machine learning methodologies perform a vital role in classifying network traffic and update the database if required. Similarly, attackers use different network classifiers to identify various applications and protocols. The performance of the classifier depends on the accuracy of collected information. Network traffic obfuscation is a technique where network traffic is manipulated (e.g. add dummy bytes with the packets in order to increase packet size) to limit the attacker's gathering of information by causing errors in the classification models. This obfuscation approach is effective at reducing the risk of passive reconnaissance where an attacker gathers traffic and uses statistical analysis to categorize different patterns (e.g. protocols, applications, user's information, etc.). A major issue of this approach is to determine the optimal algorithm for masking the features of the traffic effectively, but within the constraints of feasible modifications and limited resources or network overhead.

We propose solving this problem using Adversarial Machine Learning (AML), where a defender seeks to protect the network from an adversary by finding realistic small perturbations that are added to the network traffic to reduce the accuracy of machine learning traffic classifiers. Our contributions are three-fold:

– We introduce the Restricted Traffic Distribution Attack (RTDA), an algorithm for realistic adversarial traffic generation that can be applied in real-world networks.
– Our attack achieves state-of-the-art performance compared to previous approaches.
– We calculate the average perturbation cost for a real system and provide a comparative analysis between our proposed approach and previous work.

2 Motivation and Related Work

There are numerous reasons why network administrators need to use traffic obfuscation. For example, sometimes various internet resources are inaccessible due to an unavoidable circumstance but an administrator may want to meet performance benchmarks by shaping the network traffic. Encryption and mimicry are two basic obfuscation methods but they can not remove fingerprints from meta-data (e.g packet size, inter-arrival timing, etc.). Therefore, an adversary can classify encrypted traffic based on statistical features including packet and payload byte counts [14, 6]. Using mimicry it is possible to shape a protocol to look like another, but statistical fingerprints of meta-data are still preserved [6, 22]. We consider several data obfuscation methods that can be applied to network traffic. Our goal is to find more robust solutions for network administrators or defender in performing statistical obfuscation while minimizing unnecessary overhead using Adversarial Machine Learning (AML).

Several previous articles have proposed network obfuscation systems. Encryption and adding padding in traffic features at a variety of levels such as ciphertext formats, stateful protocol semantics, and statistical properties are effective ways of preventing statistical traffic analysis [7, 21] . Guan et al. [10] show that sending dummy traffic with real traffic (called packet padding) can manipulate an adversary's observation to a particular traffic pattern and efficiently camouflage network traffic. However, this approach is usually inefficient and sometimes incurs immense network overhead. Anjum et al. [1] use fake flows to invalidate passive reconnaissance of an adversary and also propose a non-zero-sum game-theoretic model to deploy fake flow optimally which potentially reduces network overhead and confuses adversaries in identifying network vulnerabilities. Another approach is to pad real packets to make them uniform size instead of creating a dummy packet, but it can also delay packet transmission. Wright et al. [25] proposes a convex optimization algorithm to modify real-time VoIP and WEB traffics which is optimal in terms of padding cost and reduces the accuracy of different classifiers. Later, Ciftcioglu et al. [4] propose a water-filling optimizing algorithm for optimal chaff-aided traffic obfuscation where packet morphing is performed by either chaff byte or chaff packet and show that the algorithm can maximize obfuscation given a chaff budget.

Machine learning techniques are quite common to classify the various types of IP traffics [16]. Bar-Yanai et al. [2] presents a classifier that is robust to the statistical classification of real-time encrypted traffic data. Mapping network traffic from different applications to the preselected class of services (COS) is still a challenging task. One approach uses predetermined statistical application signatures which are associated connections, sessions, application-layer protocols to determine COS class for particular datagrams [22, 5]. Zander et al. [26] use unsupervised machine learning technique to classify unknown and encrypted network protocols where flows are classified based on their network characteristics. Though classification methods are effective for statistical traffic analysis, many machine learning algorithms are vulnerable to adversarial attacks. An attacker can generate adversarial samples by adding small perturbation to the original inputs intent to mislead machine learning models [9, 8]. They can also train their own model with adversarial samples and transfer the samples to victim model in order to produces incorrect output by the victim classifier [25]. Currently, no method is effective against adversarial examples [19, 11, 17]. Papernot et al. [19] introduces adversarial sample crafting techniques that can exploit adversarial sample transferability across many of the machine learning space. There are also several mathematical and ML methods for crafting adversarial example which can exploit the gradient of the loss function or the target of classification [3, 23, 9, 20, 18]. Verma et al. [24] proposed several loss functions and the "Carlini-Wagner L_2" (also called CW) algorithm to craft network traffic using a post-processing operation to the generated distributions. However, the proposed approach sometimes created invalid perturbations and distributions for each attack that does not match real-world settings. In our work, we impose more generalized constraints in generating adversarial network traffic samples; we generate a valid perturbation and distribution for every test sample that results in a more robust attack compare to previous work.

3 Experimental Setup

This section describes the classification model and dataset, building on the previous work in [24]. In Section 5 we discuss our proposed approach in detail. Also, the table 1 shows the important notations used in this paper.

Table 1. Important Notations

Notations	**Description**
τ	Original traffic
τ_θ	Modified traffic
δ	Perturbation amount
f_θ	Classification model
ρ	Application class set
x	Feature vector
x^{adv}	Adversarial feature vector
L_p	Distance metric

3.1 Dataset

We perform experiments on the Internet Traffic Network dataset used in [15]. This dataset was generated by monitoring a research-facility host with 1000 users connected via Gigabit Ethernet link. The objects to classify are traffic flows that represent the flow of one or more packets between the host and client during a complete TCP connection. Each flow was manually classified. Table 2 shows the class information, flow types per class, and flow count. Similarly to [24], we only include the classes with at least 2000 samples in our training set.

Classification	Flow Type	Number
Bulk	FTP	11539
Database	postgres, sqlnet, oracle, ingres	2648
Mail	imap, pop2/3, smtp	28567
Services	X11, dns, ident, ldap, ntp	2099
P2P	KaZaA, BitTorrent, GnuTella	2094
WWW	www	328091

Table 2. Class composition and number used in this work from the dataset.

3.2 Realistic Features

Each sample is composed of 249 features that were observed during generation time. In a real time traffic transmission, the defender only has the capability to increase the size of the packets. Therefore, we do not use inter arrival time as a feature. Our work shows that only using packet size is sufficient to attack a network. We select the 0, 25, 50, 75, 100 percentiles of the IP packets sizes from both client-to-server and server-to-client. We normalized these features to the range $(0, 1)$.

3.3 Classification Model

We replicate the training approach and neural network model used in the previous work [24]. The training model is a 3-layer neural network with 300, 200, and 100 hidden units and applies a rectified linear function in every layer. We process the data by randomly dividing it into three datasets—5000 validation samples, 5000 test samples, and the remaining samples as training. Due to large class imbalance, we randomly sample the training set so every class has equal number of examples. We train the network using mini-batches of size 1000 for 300 epochs. The results are found in Table 3.

Class	Accuracy
Bulk	95%
Database	97%
Mail	95%
P2P	96%
Service	85%
WWW	91%

Table 3. Neural network accuracy per class.

4 Adversarial Settings

We now formalize the models for the defender and the attacker. We also discuss some well-known approaches for generating adversarial examples.

4.1 Defender Model

We model the problem by considering an adversarial setting where a defender (d) tries to protect a network from an adversary (α). The goal of α is to observe d´s network and classify its traffic flows (τ) by using statistical analysis, while d disguises τ by changing features. The new modified flow τ_θ can potentially lead α to misclassify τ_θ as relating to a different application or protocol class (σ) rather than the true one (ρ). We consider that d knows the attacker model f and observations O for training that implies d is capable to create a substitute model f_θ for τ_θ. The transferability property of AML supports that any adversarial example that can fool a machine learning algorithm can also fool other machine learning algorithms irrespective of the implementation [19]. Therefore, d uses AML techniquse to find an optimal way for generating τ_θ by considering that the traffic recipient has mechanisms for inverting the changes. However, in adding perturbations d must adhere to the following constraints:

- Basic rules of a protocol must be preserved such as packet size and timing can not be the negative, minimum or maximum range of size, etc.
- The network is be constrained by performance benchmarks meaning that the network supports a maximum threshold of latency
- The AML model should use small input perturbations for creating τ_θ since large alteration of τ can break down basic protocols and incur unnecessary network overhead

4.2 Adversary Model

We assume that α observes a particular flow between a source and destination where the flow is always bidirectional. It also has the required tools to analyze meta statistical signatures (e.g. packet size) and trains its classifier f_θ based on these features — 0, 25, 50, 75, 100 percentiles of the IP packets in both directions. The objective of α is to correctly classify the application set $\{\rho_1, \rho_2, ..., \rho_n\}$ observed in d´s traffic τ where n is the possible number of classes . Therefore, α determines a probability distribution over n classes by using $f_\theta(x)$ where x is a feature vector of $\{x_1, x_2, ..., x_n\}$ obtained from O.

4.3 Obfuscation Approaches

Let $C(x)$ be the classification of x by a model and $C^*(x)$ be the true class. Then adversarial learning finds a perturbation δ such that when added to an input x, $C^*(x) \neq C(x + \delta)$. The value of δ should be small enough when added to x for producing $x^{adv} = x + \delta$ which implies that the difference between x^{adv} and x should be almost imperceptible. While many approaches are common for generating adversarial examples, Szeged et al. [23] uses the L-BFGS optimization procedure for generating an adversarial example x^{adv} when input x is given and formulates the problem as:

$$min||x - x^{adv}||_2 + \lambda J(f_\theta(x^{adv}), t^{true})$$

The first term sets the penalty for large perturbations to x and the second one penalizes when classification deviates from the target class t^{true}. The loss function between t^{true} and output of the classifier $f_\theta(x^{adv})$ is denoted by J. $\lambda > 0$ is the model parameter.

The Carlini-Wagner L_2 attack is a robust iterative algorithm that creates adversarial examples with minimum perturbation [3]. This attack for a target class t is formalized as:

$$min||\tfrac{1}{2}(tanh(w)+1)-x||_2+\lambda f_\theta(\tfrac{1}{2}(tanh(w)+1)$$
$$\text{such that } \; C^*(x)\neq t$$

where, f_θ is defined by

$$f_\theta(x^{adv})=max(max\{Z(x^{adv})_i : i\neq t\}-Z(x^{adv})_t,-k)$$

and $\delta=\frac{1}{2}(tanh(w)+1)-x$ is the perturbation of the adversarial sample. Here, λ is chosen empirically through binary search and k controls the confidence of misclassification occurrence.

For generating untargeted adversarial perturbations Goodfellow et al. [9] proposed a fast single-step method. This method determines an adversarial perturbation under L_∞ norm where the perturbation is bounded by the parameter ϵ that results in the highest increase in the linearized loss function. It can be obtained by performing one step in the gradient sign's direction with step-width ϵ

$$x^{adv}=x+\epsilon\ sign(\mathit{\Delta}_x J(f_\theta(x^{adv}),t^{true}))$$

Here, L_∞ computes the maximum change to any of the coordinates:

$$||x-x^{adv}||_\infty=max(|x_1-x_1^{adv}|,|x_2-x_2^{adv}|,......,|x_n-x_n^{adv}|)$$

In [3], Szeged et al. used L_∞ distance metrics to generat CWL_∞ attack where the optimization functions is defined by following:

$$\lambda\ minf_\theta(x+\delta)+||\delta||_\infty$$

and, $\delta=\frac{1}{2}(tanh(w)+1)-x$. This method has a lower success rate but it is simple and computationally efficient [13].

5 Restricted Traffic Distribution Attack

We define an attack that can be translated more readily in a real-life setting. To ensure the perturbation yields a valid distribution, we have constrained our attack in two ways: the attack is not allowed to reduce the packet size, and the generated distribution should preserve the monotonic non-decreasing property. We solve this problem by enforcing these constraints directly in the adversarial optimization framework.

Notice that it is possible to reduce the packet size in a distribution by inserting small dummy packets into the traffic, but this approach introduces a larger overhead into the network than only appending dummy bytes.

5.1 Perturbation constraints

Given a distribution x, a general adversarial algorithm finds a perturbation δ that it minimizes a distance metric L_p and changes the correct classification: $C^*(x) \neq C(x+\delta)$. This perturbation has no restrictions with respect to the direction that modifies the original distribution. Instead, we clip every value below zero in the perturbation during learning:

$$\begin{array}{rl} \text{minimize} & L_p(x, x+(\delta)^+) \\ \text{such that} & C(x+(\delta)^+) = t \end{array}$$

where $(f)^+$ stands for $max(f,0)$ and t is not the correct label.

5.2 Distribution constraints

Given a batch of adversarial distributions A, we define an operation that identifies every adversarial sample with decreasing consecutive features.

Let (A_i, A_{i+1}) be consecutive features in the batch A, we compute the following operation:

$$diff := (A_i - A_{i+1})^+$$

We update our distribution based on this value: $A_{i+1} := A_{i+1} + diff$. For a valid sample the operation will result in 0, but for an invalid one it will compute the difference between features, so after the update we automatically get non-decreasing features. This operation is sequentially applied to every pair of consecutive features in the same distribution during the optimization of the attack.

5.3 Framework

These restrictions in an attack should generate a valid adversarial distribution if convergence is possible. In this work we choose the Carlini-Wagner attack for L_2 and L_∞ norm as our frameworks.

Implementation Details. We re-implement the Carlini-Wagner attack for L_2 and L_∞ norm. For the initial c we select 10^{-3} and 10^{-1}, respectively, and search for 5 steps with 1000 as the maximum number of iterations. In our algorithm, we clip the perturbation before adding to the batch, and then apply the series of operations to correct the distribution. We also replicate the method reported in [24] by applying a post-processing operation to the generated distributions from CW L_2 attack.

	Valid perturbation	**Valid distribution**
RTDA CW L_2	100%	100%
Post-processing CW L_2	0%	100%
CW L_2	0%	20%
RTDA CW L_∞	100%	100%
CW L_∞	0%	22%

Table 4. Percentage of valid adversarial samples.

6 Results

We test our two RTDA frameworks against previous adversarial approaches.

We compare the attacks by evaluating how realistic the generated distributions are, and the success rate for fooling the neural network. To evaluate how realistic an attack is we compare the ratios of valid perturbations and valid distributions for each attack. The results are shown in Table 4 for realistic attacks and Table 5 for success rate per class.

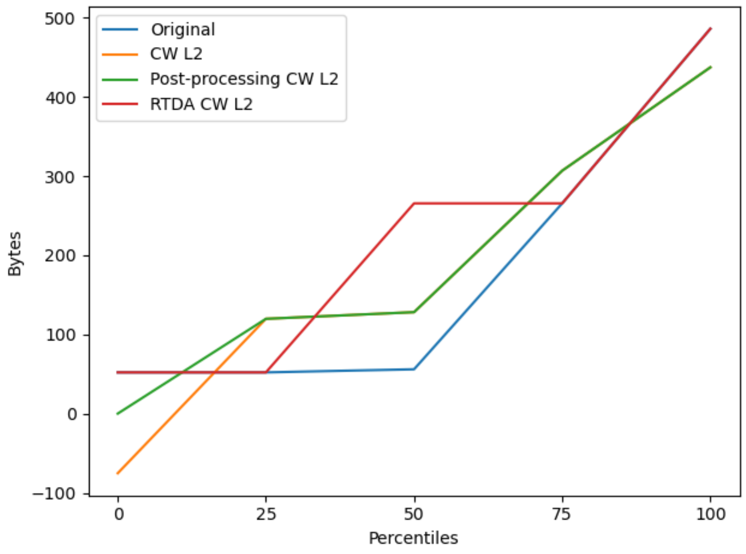

Fig. 1. A comparison of every L_2 adversarial example generated from the same distribution. Notice the negative packet size generated by CW L_2 and the reduction of 0th and 100th percentile by post-processing CW L_2. In contrast, RTDA generates an adversarial by only increasing the 50th percentile.

Prior work did not consider the limitations of a perturbation in real-world settings. Our algorithm is significantly more realistic than previous attacks. Both

	Database	**Bulk**	**Mail**	**P2P**	**Services**	**WWW**
RTDA CW L_2	100%	100%	95%	93%	100%	95%
Post-processing CW L_2	75%	33%	29%	50%	53%	84%
CW L_2	100%	100%	100%	100%	100%	100%
RTDA CW L_∞	100%	100%	74%	72%	67%	100%
CW L_∞	100%	100%	100%	100%	100%	100%

Table 5. Success rate per class (Fraction of instances for which an adversarial was found).

of our frameworks generate a valid perturbation and distribution for every test sample. Post-processing has the disadvantage that resultant distributions may no longer be adversarial examples. Our approach directly finds attacks in the valid space allowing to optimize towards the best attacks. Therefore, RTDA outperforms significantly the success rate of the previous post-processing CW L_2 attack in every class. Even with the additional constraints our attack L_2 and L_∞ are just 2% and 14% apart respectively from their unrestricted versions.

	L_p	δ **mean (Bytes)**
RTDA CW L_2	0.015	14.5
Post-processing CW L_2	0.012	15.4
CW L_2	0.011	12.9
RTDA CW L_∞	0.033	19.41
CW L_∞	0.026	15.74

Table 6. Average norm and perturbation.

Both attacks have a larger norm in comparison to previous approaches. Surprisingly, RTDA L_2 has a smaller perturbation than the post-processed approach. On average, our attack can be applied to a system by increasing each packet by 14.5 bytes. Table 6 compares the corresponding norm and average perturbation for each approach.

7 Conclusions and Future Work

Network traffic is vulnerable to statistical analysis in which an adversary can classify various types of applications and protocols by observing unencrypted meta signature of network packets. Adversarial machine learning techniques are very effective for obfuscating network traffic while introducing minimal network overhead. We proposed a novel network traffic obfuscating approach that is robust against network traffic attackers where we leverage adversarial attacks as a mechanism to obfuscate network traffic. Our algorithm outperforms previous approaches achieving state-of-the-art results and reduces the network overhead produced by the perturbation.

We plan to extend this work by testing our approach in a real world network and/or network traffic simulators. In addition, we are working on generating adversarial examples by using game-theoretic models where the defender adds various perturbation to the original features of different classes by paying variable cost to confuse the attacker in decision making. This game model seeks to simulate the AML approach's settings by initially imposing two general constraints—a positive perturbation and total perturbation bounded by a cost budget. We expect that similar approaches would also be effective in introducing other realistic constraints into the model, allowing simple but robust perturbations to limit traffic classification accuracy and reconnaissance value.

8 Acknowledgment

This work was supported by the Army Research Office under award W911NF-17-1-0370.

References

1. Anjum, I., Sujan Miah, M., Zhu, M., Sharmin, N., Kiekintveld, C., Enck, W., Singh, M.P.: Optimizing vulnerability-driven honey traffic using game theory. arXiv pp. arXiv–2002 (2020)
2. Bar-Yanai, R., Langberg, M., Peleg, D., Roditty, L.: Realtime classification for encrypted traffic. In: International Symposium on Experimental Algorithms. pp. 373–385. Springer (2010)
3. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 ieee symposium on security and privacy (sp). pp. 39–57. IEEE (2017)
4. Ciftcioglu, E., Hardy, R., Chan, K., Scott, L., Oliveira, D., Verma, G.: Chaff allocation and performance for network traffic obfuscation. In: 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS). pp. 1565–1568. IEEE (2018)
5. Duffield, N.G., Roughan, M., Sen, S., Spatscheck, O.: Statistical, signature-based approach to ip traffic classification (Feb 9 2010), uS Patent 7,660,248
6. Dusi, M., Crotti, M., Gringoli, F., Salgarelli, L.: Tunnel hunter: Detecting application-layer tunnels with statistical fingerprinting. Computer Networks **53**(1), 81–97 (2009)
7. Dyer, K.P., Coull, S.E., Shrimpton, T.: Marionette: A programmable network traffic obfuscation system. In: 24th USENIX Security Symposium (USENIX Security 15). pp. 367–382. USENIX Association, Washington, D.C. (Aug 2015), https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/dyer
8. Goodfellow, I., Papernot, N., McDaniel, P., Feinman, R., Faghri, F., Matyasko, A., Hambardzumyan, K., Juang, Y.L., Kurakin, A., Sheatsley, R., et al.: cleverhans v0. 1: an adversarial machine learning library. arXiv preprint arXiv:1610.00768 **1** (2016)
9. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)

10. Guan, Y., Fu, X., Xuan, D., Shenoy, P.U., Bettati, R., Zhao, W.: Netcamo: camouflaging network traffic for qos-guaranteed mission critical applications. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans **31**(4), 253–265 (2001)
11. He, W., Wei, J., Chen, X., Carlini, N., Song, D.: Adversarial example defense: Ensembles of weak defenses are not strong. In: 11th {USENIX} Workshop on Offensive Technologies ({WOOT} 17) (2017)
12. Karthika, C., Sreedhar, M.: Statistical traffic pattern discovery system for wireless mobile networks. Computer Science & Telecommunications **45**(1) (2015)
13. Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial examples in the physical world. arXiv preprint arXiv:1607.02533 (2016)
14. Mohajeri Moghaddam, H., Li, B., Derakhshani, M., Goldberg, I.: Skypemorph: Protocol obfuscation for tor bridges. In: Proceedings of the 2012 ACM conference on Computer and communications security. pp. 97–108 (2012)
15. Moore, A.W., Zuev, D.: Internet traffic classification using bayesian analysis techniques. In: Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems. pp. 50–60 (2005)
16. Nguyen, T.T., Armitage, G.: A survey of techniques for internet traffic classification using machine learning. IEEE communications surveys & tutorials **10**(4), 56–76 (2008)
17. Ortiz, A., Fuentes, O., Rosario, D., Kiekintveld, C.: On the defense against adversarial examples beyond the visible spectrum. In: MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM). pp. 1–5. IEEE (2018)
18. Ortiz, A., Granados, A., Fuentes, O., Kiekintveld, C., Rosario, D., Bell, Z.: Integrated learning and feature selection for deep neural networks in multispectral images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 1196–1205 (2018)
19. Papernot, N., McDaniel, P., Goodfellow, I.: Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint arXiv:1605.07277 (2016)
20. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: 2016 IEEE European symposium on security and privacy (EuroS&P). pp. 372–387. IEEE (2016)
21. Pinheiro, A.J., Bezerra, J.M., Campelo, D.R.: Packet padding for improving privacy in consumer iot. In: 2018 IEEE Symposium on Computers and Communications (ISCC). pp. 00925–00929 (2018)
22. Roughan, M., Sen, S., Spatscheck, O., Duffield, N.: Class-of-service mapping for qos: a statistical signature-based approach to ip traffic classification. In: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement. pp. 135–148 (2004)
23. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
24. Verma, G., Ciftcioglu, E., Sheatsley, R., Chan, K., Scott, L.: Network traffic obfuscation: An adversarial machine learning approach. In: MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM). pp. 1–6. IEEE (2018)
25. Wright, C.V., Coull, S.E., Monrose, F.: Traffic morphing: An efficient defense against statistical traffic analysis. In: NDSS. vol. 9. Citeseer (2009)
26. Zander, S., Nguyen, T., Armitage, G.: Self-learning ip traffic classification based on statistical flow characteristics. In: International Workshop on Passive and Active Network Measurement. pp. 325–328. Springer (2005)