Rapid: Robust and Adaptive Detection of Distributed Denial-of-Service Traffic from the Internet of Things

Samuel Mergendahl
University of Oregon
Eugene, OR
smergend@cs.uoregon.edu

Jun Li
University of Oregon
Eugene, OR
lijun@cs.uoregon.edu

***Abstract*—The ubiquity of the Internet of Things (IoT) can cause devastating distributed denial-of-service (DDoS) attacks. In order to thwart IoT-enabled DDoS attacks, network operators can deploy an anomaly detection system that detects and mitigates the DDoS traffic near the source of the attack. Unfortunately, anomaly detection systems face conflicting real-world deployment challenges in IoT networks. Namely, many IoT network operators cannot deploy systems that require manual parameter tuning or labeling, but without such system maintenance, previously proposed anomaly detection systems display far too many false positives for acceptable deployment in IoT networks. To exacerbate this problem, pre-trained machine learning models often fail to accurately classify traffic when ported to a new environment (i.e., domain shift), and must either retrain with labeled data painstakingly collected from the newly deployed network or exhibit poor accuracy.**

In order to overcome these IoT deployment challenges, we propose a novel anomaly detection system, Rapid, that successfully detects IoT-enabled DDoS attacks but also meets the accuracy demands of IoT under domain shift with little to no system maintenance from IoT network operators. Rapid initially provides robust detection through advanced machine learning techniques such as Long Short Term Memory (LSTM), but further, Rapid introduces a novel active learning technique that interweaves closely with attack mitigation to automatically adapt to new network domains without manual system maintenance. Rapid operates in real-time, and even though we leverage neural network techniques that typically cannot explain their predictions, our system architecture can provide diagnostic insight into any detected attack.

I. Introduction

The ubiquity of the Internet of Things (IoT) continuously disrupts the infrastructure of the Internet in unintended ways. As non-security professionals often act as the network operator for IoT networks (e.g., many smart-home networks), poor security practices such as default login passwords for the IoT devices within the network are common. These IoT devices consequently become compromised and collectively form botnets that can launch devastating distributed denial-of-service (DDoS) attacks. In fact, due to the ubiquity of these IoT botnets [1], [2], networks desperately need a means to detect and mitigate modern IoT-enabled DDoS attacks [3], [4].

Anomaly detection plays a pivotal role in the detection and mitigation of IoT-enabled DDoS attacks. With the assumption that malicious traffic appears statistically different from benign traffic, anomaly detection defines a normal behavior profile for benign traffic and classifies any traffic statistically outside the normal behavior profile as anomalous. A network operator can define the classification boundaries between anomalous and normal traffic through a manual statistical investigation of previous traffic in which the network operator selects static classification thresholds, a machine learning algorithm that trains on previous traffic and automatically derives classification thresholds, or more advanced machine learning techniques, such as neural networks, that detect anomalous traffic through a black-box approach (i.e., the network operator only learns traffic labels not the classification boundary). Of course, unsupervised machine learning techniques can extract anomalies in a static set of data, but in order to discern the malevolence of future traffic, even unsupervised methods must compare the future traffic to an extracted classification boundary (i.e., we consider this process training). In particular, anomaly detection can discover a more diverse set of attacks on future traffic than signature-based detection alone. Moreover, when deployed at the source-end (i.e., near the source of the attack), a properly constructed anomaly detection system can pair with a mitigation system to prevent complex IoT-enabled DDoS attacks that victim-end defenses cannot [5].

Unfortunately, anomaly detection faces five critical real-world deployment challenges in the detection of IoT-enabled DDoS attacks. First, because IoT devices often maintain strict energy consumption requirements, the high cost of errors caused by anomaly detection intensifies in IoT networks [6]. Namely, anomaly detection traditionally exhibits high false positive rates, and a false positive for IoT-enabled DDoS will trigger a mitigation system to drop benign traffic which leads to increased retransmission and energy consumption for benign IoT devices [7]. Second, because many IoT networks deploy through non-security professionals, the anomaly detection system must meet an even higher bar for ease of deployment than the already demanding requirements of typical network operators [8]. Without a security professional to analyze network traffic and manually tune classification thresholds, a pre-trained machine learning-based anomaly detection scheme becomes the only reasonable design choice. Third, because of the heterogeneity of IoT, pre-trained anomaly detection systems exhibit magnified false positives under domain shift [9]. The normal traffic profile that an anomaly detection system defines for one IoT network may have little relevance for another IoT network.

Fourth, because of IoT often directly connects with the physical world, any automated decision from an anomaly detection system must allow a human-in-the-loop to make structured configuration changes. For example, an unexplainable system is unacceptable in many IoT environments, such as Industrial Control Systems (ICSes) [10]. Finally, due to the computational constraints of many IoT devices, the chosen anomaly detection algorithm must appropriately weigh system complexity and accuracy with real-time operation [11].

To exacerbate these challenges, a solution to solve one specific IoT deployment challenge often fundamentally neglects or contradicts a different IoT deployment challenge. For example, a pre-trained neural network is a powerful machine learning technique that may meet the accuracy demands of IoT, but fails to provide an explainable detection system. Similarly, in order to combat domain shift, a network operator can leverage active learning techniques in which a pre-trained machine learning model labels a pool of unlabeled data from the policed network. Thus, before deployment in the policed network, the pre-trained system can avoid domain shift by retraining on labeled data from the policed network. However, as the pre-trained model may sometimes exhibit low confidence on the unlabeled pool (e.g., a classification probability near 0.5), domain shift requires a security professional to assist in the active learning process and manually label any data the model cannot classify with high confidence.

We propose a new anomaly detection scheme, **Rapid**, that overcomes each of the aforementioned challenges. Rapid deploys at the source-end and detects any outgoing IoT-enabled DDoS attacks, but most importantly, Rapid departs from previous work with a focus on real-world deployability in IoT networks. First, in order to achieve **state of the art accuracy** for detection of IoT-enabled DDoS attacks, we leverage a Multilayer Perceptron (MLP) to ensemble the output of various statistical methods, and further, we deploy a Long Short Term Memory (LSTM) architecture to take full advantage of the sequential nature of network data. Second, because IoT network operators require an **easily deployable** system, our detection architecture automatically tunes every necessary parameter, and we provide a pre-trained model ready for deployment in any IoT network. Third, as the accuracy of pre-trained models suffer under domain shift, Rapid **automatically adapts to any network** through novel active learning techniques that interweave closely with attack mitigation rather than require manual effort from a security professional. Fourth, with our careful feature selection, even though Rapid utilizes a neural network architecture that typically cannot explain its predictions, our design **provides diagnostic insight** into any detected attacks. Finally, we deploy a cost efficient architecture and implementation that **operates in real-time** and achieves online classification.

We organize the remainder of this paper as follows. Section II reviews the related work on previous anomaly detection systems and discusses the deployment challenges for such defenses. Section III introduces the design of our anomaly detection system which addresses the major deployment concerns of previous anomaly detection systems in IoT environments. Section IV provides a comprehensive evaluation of our anomaly detection system that uses multiple real-world IoT traffic traces for a realistic simulation environment in which we compare our system to the current state of the art anomaly detection schemes. Finally, Section V concludes this work.

II. Related Work

Even though manual statistical analysis for anomaly detection was first proposed many years ago, the statistical technique, data separation, is still an active research area [11], [12], [13], [14], [15]. Data separation performs mathematical transformations on the traffic features to accentuate anomalous behavior. In particular, the most common technique for data separation is Principal Component Analysis (PCA) [11], [15]. For example, Xie et al. use an orthogonal transformation to convert possibly correlated observed variables into a set of linearly uncorrelated variables called Principle Components (PCs), and whenever the variance between sequential PCs surpasses a threshold, the defense flags an anomaly [11]. While these simple statistical detection schemes operate with low system overhead, they suffer from high rates of false positives [6], and a false positive in IoT-enabled DDoS leads to dropped benign traffic and increased energy consumption for benign IoT devices [7]. Moreover, success of these systems rely on a correct classification threshold that non-security professionals in IoT networks cannot determine manually.

Rather than manually set classification thresholds, machine learning algorithms automatically derive the classification boundaries between normal traffic and anomalous traffic [8], [16], [17], [18], [19]. In particular, due to their superior accuracy, recent machine learning-based anomaly detection systems choose to deploy neural network architectures [18], [19]. For example, in order to define the classification boundary between normal and anomalous traffic, Mirsky et al. introduce Kitsune, an ensemble of autoencoders—a neural network designed to reconstruct its inputs—that efficiently learns complex behaviors and traffic patterns of benign traffic [19]. While an improvement for accuracy, Kitsune acts as a black-box to the network operator and cannot explain any anomalous behavior. Furthermore, even a pre-trained model of Kitsune will suffer under domain shift in IoT networks without a network operator to manually label anomalous traffic.

To properly handle domain shift, a pre-trained anomaly detection system must retrain on labeled data from the policed network, but manually labeling traffic within the policed network is a painstaking process. Thus, recent work attempts to limit the effort required of network operators to obtain labeled data from their network [9], [20]. For example, Beaugnon et al. present a system, ILAB, that leverages active learning techniques along with a pre-trained machine learning model to label a pool of target data [20]. When the machine learning model cannot confidently label a data point, ILAB requests a security professional to manually label the traffic data in question. ILAB significantly limits the manual effort of a network operator to

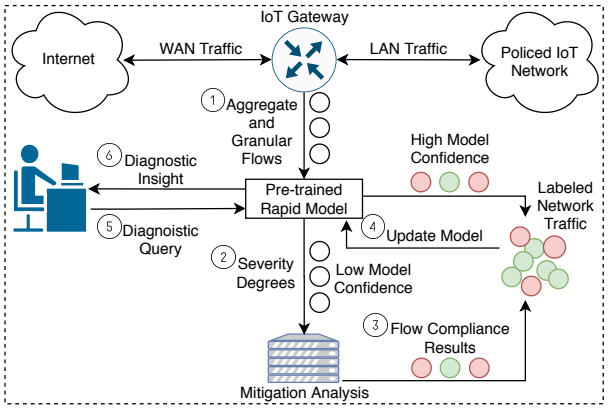


Fig. 1: Rapid's gateway infrastructure.

label traffic, but many IoT network operators may not possess the required skill to manually label even limited data points.

While each of the anomaly detection systems above can directly deploy in IoT networks, other anomaly detection systems specifically tailor their design for IoT [10], [21], [22], [23], [24]. In particular, similar to Rapid, Feng et al. [10] and Linda et al. [24] build their anomaly detection systems with Long Short Term Memory (LSTM) for Industrial Control Systems (ICS) and a Multilayer Perceptron (MLP) for Critical Infrastructure (CI) respectively. However, while each system meets the accuracy demands of IoT, neither system can explain their detection results. In fact, Feng et al. discuss this issue and deploy a bloom filter along side their LSTM [10] which provides side information about the environment in which the neural network operates. However, since the LSTM does not use the statistical method directly as a feature, the network operator cannot guarantee the relevance of the statistical method to the classification decision. Furthermore, Hamza et al. build an anomaly detection system designed to detect IoT-enabled DDoS attacks [21]. Similar to Rapid, the authors use a machine learning-based anomaly detection to identify IoT-enabled DDoS attacks, but unlike Rapid, the authors cannot guarantee their pre-trained model will operate correctly under domain shift. In fact, we train Rapid on a similar dataset as Hamza et al. [21] to illuminate the shortcomings of their design.

III. System Design

A. System Overview

We first assume that our anomaly detection system resides at the gateway of a generic IoT network with access to traffic that enters and leaves the IoT network. Due to the heterogeneity of IoT, we make no assumptions on the type of IoT network that deploys our system except for the resource constrained nature of its end-hosts. As physical access to many IoT devices can prove difficult (e.g., underwater networks or sensors within a nuclear reactor), we demand no hardware or software requirements of the IoT devices. For example, the network can exhibit traits of a Wireless Sensor Network (WSN), Mobile Ad-hoc Network (MANET), or Intermittently Connected Network (ICN), and/or deploy in a smart-home, healthcare facility, large-scale factory, or Cyber Physical System (CPS).

Our defense must detect any IoT-enabled DDoS traffic that attempts to leave the IoT network. Specifically, we aim to detect any volumetric DDoS attack that originates within the policed IoT network and targets an external victim. While our architecture maintains the intrinsic capability to detect internal DDoS attacks, application layer DDoS attacks, flash-crowds, and external DDoS attacks launched toward the IoT network, we leave such attacks formally outside the scope of this work.

There are five main design goals Rapid must meet. First, in order to achieve state of the art performance, we leverage a classification architecture based on Long Short Term Memory (LSTM). Second, Rapid must easily deploy in an IoT network, so we preemptively tune every parameter in our classification architecture and provide a pre-trained model for any IoT network. Third, as any pre-tuned anomaly detection system inevitably performs sub-optimal under domain shift, Rapid must *automatically* retrain with relevant labeled traffic whenever the system deploys in a new environment. Fourth, in order to provide diagnostic insight into classifications, we only leverage neural networks to ensemble a set of statistical methods. Finally, we must maintain a real-time classification system, so our architecture includes a streamlined feature extraction process, a suite of performant statistical analysis, and an optional DDoS attack detection component to limit the computational overhead of the classification system. See Fig. 1 for an overview of Rapid.

B. Feature Extraction and Statistical Analysis

Because Rapid can classify IoT-enabled DDoS traffic without the use of payload information, we can efficiently mirror the traffic at the gateway with sFlow [25]. sFlow, or sampled flow, is an industry standard for packet export that can handle high traffic links. In order to limit the computational overhead of Rapid, Rapid can change the sampling rate of sFlow (e.g., rather than analyze every single packet, Rapid can set sFlow to mirror only a fraction of packets). However, we leave a formal evaluation of the relationship between sampling rates and classification accuracy to future work. We separate collected sFlow streams into *aggregate* flows and *granular* flows. Aggregate flows represent groups of sFlow packets with the same external IP address, and granular flows represent groups of sFlow packets that share the same internal and external IP addresses. We use aggregate flows for attack detection and granular flows for attack classification. See step 1 in Fig. 1 for more details.

During every time window, we describe each flow (both aggregate and granular) with four extracted features: (1) total outgoing bytes, (2) a ratio of incoming to outgoing bytes, (3) total outgoing packets and, (4) a ratio of incoming and outgoing packets. Because early DDoS detection solutions (along with a static threshold) used these features directly as their detection system [26], [27], [28], [29], we call these features *basic detectors*. In particular, for each flow, we create an time-ordered list of the last n values of each basic detector (i.e., each flow has four associated basic detector time series). While the basic detectors fail to meet real-world standards for detection by

themselves, they offer condensed, descriptive characteristics specifically tailored for DDoS detection. Namely, these features may miss anomalies specific to attacks other than DDoS attacks (e.g., worm detection), but as Rapid focuses solely on the detection of DDoS traffic, we rely on the previous studies to assume our basic detectors are appropriate features for volumetric DDoS detection.

Given each basic detector time series, we perform multiple statistical tests to understand the state of the network. Rather than directly determine the class label for each current flow at this step (as with most previous machine learning techniques for anomaly detection), our statistical methods output the extent current traffic differs from previous traffic—which we call the *severity degree*—and leave the classification decision to future components. In particular, in order to derive a set of severity degrees for each traffic flow, we analyze each basic detector time series with multiple parameterizations of the Auto-Regressive Integrated Moving Average (ARIMA) algorithm [30]. The ARIMA algorithm forecasts the next value in a time series and can provide a collection of confidence intervals (e.g., ARIMA may output 90% confidence that the next value in a series will land in the interval $[0, 100]$, 50% confidence that the next value in a series will land in the interval $[25, 75]$, etc). Thus, the severity degree each ARIMA algorithm outputs for each basic detector value corresponds to the smallest confidence interval in which the current basic detector value resides. Moreover, ARIMA uses three integer parameters, (p, d, q), that when set properly, can represent many commonly known time series approaches (e.g., ARIMA parameterized with $(0, 0, 1)$ is a simple moving average). We consider one combination of p, d, and q to represent one parameterization of ARIMA.

C. Ensembled Classification with Deep Learning

The choice of which ARIMA parameterizations to deploy depends on the specific time series. But, as an IoT network operator cannot perform manual investigations to discern which ARIMA parameterization best suites the current traffic patterns, we instead take a comprehensive approach and feed the severity degrees from the first 27 parameterizations of ARIMA, $(0, 0, 0)$ to $(2, 2, 2)$, into an ensemble algorithm that automatically determines which parameterization to follow. Unfortunately, traditional ensemble methods, such as the random forest classifier, Opprentice, used by Liu et al. [8], cannot achieve the accuracy guarantees IoT security applications require. Therefore, in order to achieve state of the art accuracy, we ensemble the collected severity degrees with two neural network techniques. First, we use a Multilayer Perceptron (MLP) [31] to ensemble the severity degrees into a single severity degree. Due to its non-linearity, the MLP can derive a more complex understanding of the severity degrees than traditional machine learning techniques (e.g., the MLP learns which ARIMA parameterization to highlight based on underlying traffic distributions). Second, we use Long Short Term Memory (LSTM) [31] to analyze the output of the MLP over many time windows and submit the final, single probabilistic severity degree of our system to the mitigation component (i.e., see step 2 in Fig. 1). Because the MLP only investigates the severity degrees of the current time period, the LSTM allows our architecture to further take into account past trends of prediction. See Fig. 2 for more details.

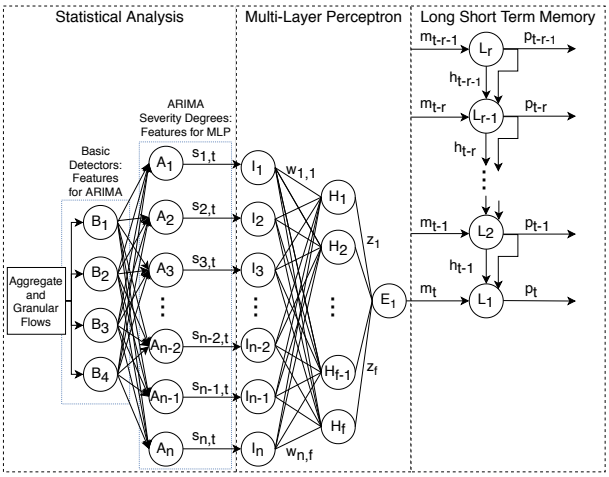


Fig. 2: Rapid's classification architecture.

With Fig. 2 as a reference, we can formally define the various aspects of our classification architecture as follows. At each time window t, the IoT gateway mirrors sFlow streams that Rapid collects into granular and aggregate flows in which the four basic detectors, or B_i, extract features for ARIMA. Each of the n parameterizations of ARIMA, or A_i, receives the four basic detector values and outputs a severity degree, $s_{i,t}$. Together, each severity degree feeds into the visible layer of the MLP such that $s_{i,t}$ enters the cell I_i in which each neuron applies the ReLU activation function. Subsequently, we apply each synapse of the MLP, $w_{i,j}$, to the value towards the hidden layer—specifically neuron H_j—until we arrive at the final output of the MLP, m_t. The LSTM then incorporates r time windows to derive the final probabilistic severity degree, p_t, for the time window t. Specifically, the hidden memory, h_{t-1}, previous severity degree, p_{t-1}, and the current MLP output, m_t, act as input to the final cell of the LSTM.

D. Domain Adaptation through Active Learning

Even though our neural network can learn many statistical subtleties of the network traffic, domain shift can skew our pre-trained model if the new domain contains *different* subtleties. Thus, every time period, Rapid retrains and retunes its parameters. Concretely, Rapid replaces a subset of the original training data with labeled data gathered from the previous time period. In order to maintain a diverse, but recent training data set, Rapid intelligently chooses which traffic to replace (e.g., if the previous time period contains no DDoS traffic, Rapid will not replace malicious training samples). Because we feed our neural network a comprehensive set of ARIMA parameterizations, even if Rapid deploys in a new IoT environment, Rapid can quickly adapt to new underlying assumptions (i.e., the neural network will learn to emphasize a new set of ARIMA parameterizations specific to the new environment).

Dataset	Benign IoT Traffic	Benign Non-Iot Traffic	UDP Flood	TCP SYN Flood	HTTP Flood	DNS Flood
Smart Home 1 [32]	✓	✓	✓	✓	✓	✗
Smart Home 2 [33]	✓	✗	✗	✗	✗	✗
Smart Hospital [34]	✓	✓	✗	✗	✗	✗
CAIDA DDoS Attack [35]	✗	✗	✗	✓	✗	✗
Booter DDoS Attack [36]	✗	✗	✗	✗	✗	✓
DARPA DDoS Attack [37]	✗	✓	✗	✓	✗	✗

TABLE I: List of real-world traffic traces used for our evaluation.

As we cannot rely on an IoT network operator to label traffic in the new domain, throughout each time period, Rapid partners with an attack mitigation system to label traffic based on compliance with the mitigation techniques. Because attack mitigation must deploy traffic engineering techniques such as throttling or filtering traffic to promptly respond to DDoS attacks, recent attack mitigation research suggests that a connection's response to these traffic engineering techniques offers further insight into the malevolence of the connection [7]. For example, if attack mitigation drops traffic in a TCP connection, and the sender fails to comply with congestion control (and continues to send at the same high rate), Rapid can label this previous traffic as malicious. Thus, attack mitigation acts as a security professional and automatically provides the labeled traffic necessary to combat domain shift. To save computational resources, Rapid only queries attack mitigation for low confidence classifications and directly labels high confidence classifications (i.e., see step 3 and step 4 in Fig. 1).

E. Diagnosing Classifications

While automation with deep learning can solve complicated problems, deep learning cannot *explain* its results. Due to the hidden layers of the neural network, network operators will struggle to discern the underlying reason for a particular classification given by the model. Compounded with the fact that real-world security applications can never guarantee perfect implementation, diagnostic insight becomes critical for a human-in-the-loop to resolve any deployment issues. For example, if an IoT application begins to exhibit poor quality of service, the network operator will likely wish to investigate why. Without a rigorous explanation, the network operator may develop a distrust for the security application and remove its deployment. Furthermore, if the security application can provide a rigorous explanation, the human-in-the-loop can pinpoint any misconfiguration of the security system, and nudge the system back to proper deployment.

While directly using the basic detectors as input to our neural network may improve our accuracy, Rapid would become unexplainable. The statistical analyses provide a means for diagnostic insight into the classifications. Because the LSTM *directly* makes its predictions based on the statistical analysis, we have assurance that the LSTM decision correlates with the severities degrees. In fact, Rapid provides a simple statistical diagnostic that the network operator can request for a particular flow (i.e., see step 5 and step 6 in Fig. 1).

F. Attack Detection

Attack detection aims to reduce the overhead of classification. We assume that, more often than not, network behavior is benign. Therefore, if the system constantly analyzes the traffic at a fine-grained scale (e.g., every granular flow), the majority of system execution is unnecessary. Thus, to save resources, by default, Rapid applies its classification methodology on aggregate flows, and only when an aggregate flow appears anomalous, Rapid analyzes each granular flow within the suspicious aggregate flow. However, as attack detection is not fundamental to the success of Rapid, but rather, an option for extremely resource constrained networks whose IoT gateway maintains strict resource consumption requirements, we allow a network operator to turn attack detection on or off depending on their situation. In fact, because attack classification maintains the final classification decision, a false positive in attack detection simply means attack classification executed unnecessarily. However, even with false positives during attack detection, attack classification will still execute less frequently than with no attack detection component. Therefore, we can sacrifice the false positive rate of attack detection to maintain very low false negatives during attack detection. Moreover, attack detection and classification can sequentially execute within a small time frame, so a network operator can add attack detection without fear of noticeable added latency for classification. We discuss the trade offs of attack detection further in Section IV.

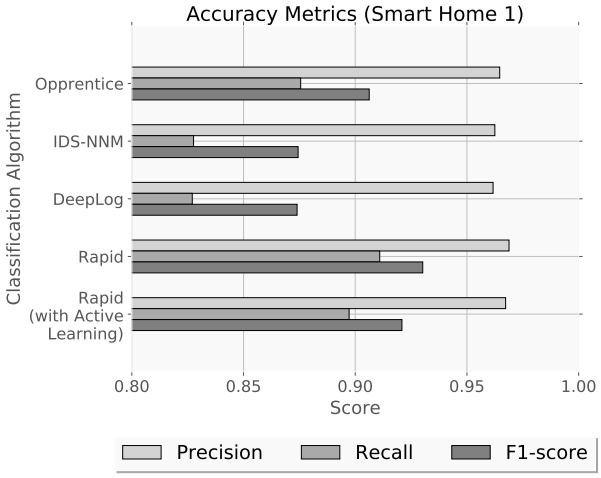
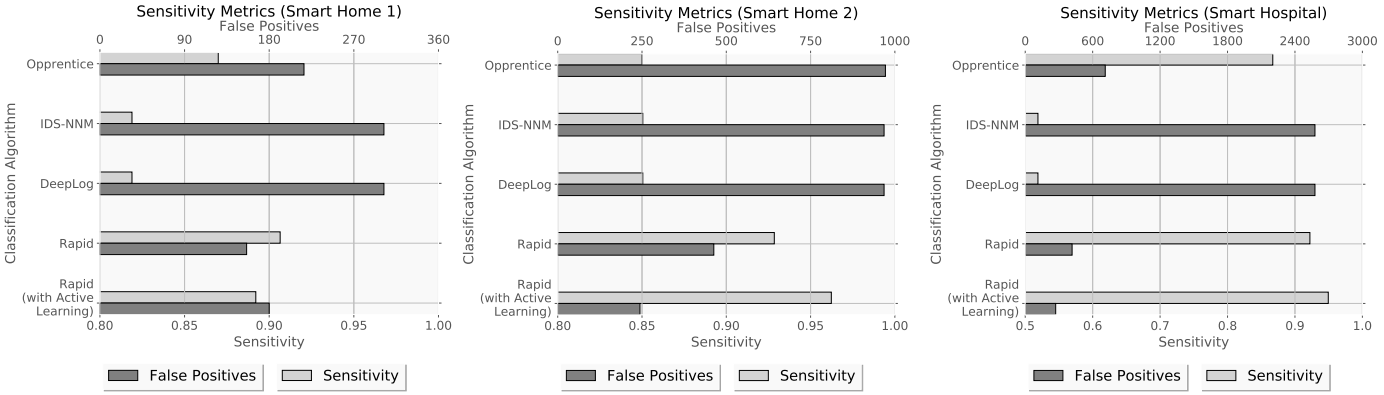


Fig. 3: Precision, Recall, and F1-score of each system.

IV. Evaluation

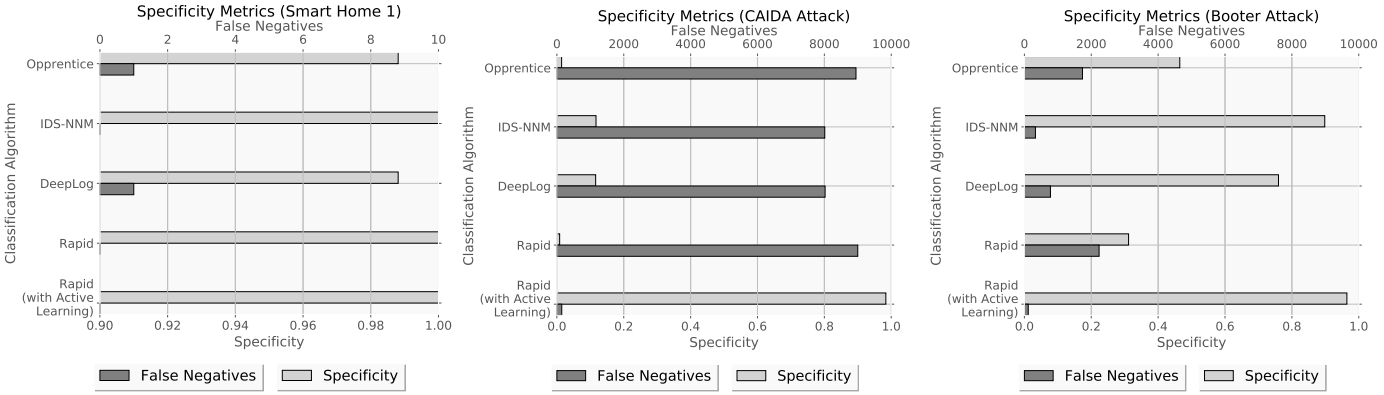
A. Datasets and Evaluation Overview

In order to distill confidence that Rapid overcomes the IoT deployment challenges, we must evaluate Rapid in a scenario as close to the real-world as possible. In particular, we select six different datasets of realistic traffic exhibited in actual networks. While no perfect dataset exists (i.e., a network trace



(a) Original domain (Smart Home 1). **(b)** Under domain shift (Smart Home 2). **(c)** Under domain shift (Healthcare Network).

Fig. 4: Sensitivity of each anomaly detection system under various environments.



(a) Original domain (Smart Home 1). **(b)** Under domain shift (CAIDA Dataset). **(c)** Under domain shift (Booter Dataset).

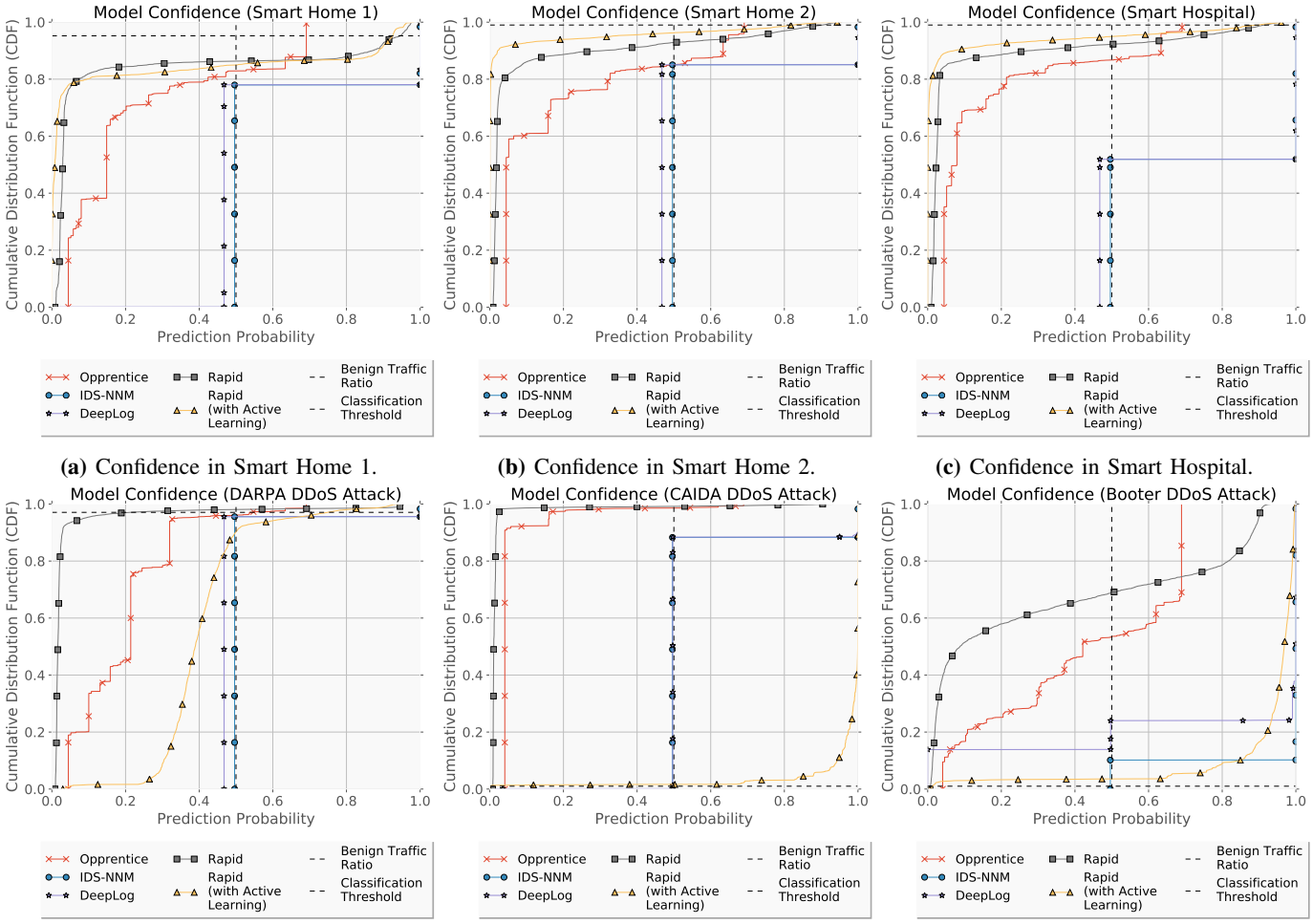
Fig. 5: Specificity of each anomaly detection system under various attack traces.

that contains benign IoT traffic, benign non-IoT traffic, and multiple non-synthetic IoT-enabled DDoS attacks), let alone multiple such datasets to study the effect of domain shift, we combine various real-world datasets for a comprehensive evaluation. For further details of each dataset, see TABLE I.

We compare Rapid to various state of the art anomaly detection systems. Namely, we evaluate Rapid against the anomaly detection algorithms seen in Opprentice [8], IDS-NNM [24], and DeepLog [18]. Opprentice ensembles various statistical methods together for a final probabilistic severity degree, but unlike Rapid, Opprentice leverages a Random Forest classifier rather than a neural network to perform the ensemble procedure. Second, the IDS-NNM system uses a Multilayer Perceptron (MLP) to identify anomalies in a supervisory control and data acquisition (SCADA) network. Lastly, to leverage the sequential nature of network events, DeepLog uses a Long Short Term Memory (LSTM)-based anomaly detection system. While IDS-NNM and DeepLog leverage neural network techniques like Rapid, both systems directly use features from the network rather than various statistical methods. Furthermore, in order to compare directly with Rapid, we focus on the classification model architecture of the previous systems and adapt the surrounding system—with potential implementation changes—to the same environment as Rapid (i.e., we test Opprentice in the security domain, we test IDS-NNM in IoT environments other than ICS, and we use network logs as input to DeepLog rather than system logs).

B. Model Accuracy

We begin with the standard evaluation of a classification system to show Rapid can classify traffic with state of the art accuracy. First, we leverage a smart-home dataset that contains both IoT benign traffic and multiple different IoT-enabled DDoS attacks [32]. In particular, we split this dataset into three main pieces: train, validation, and test data. For the remainder of our evaluations, each anomaly detection system only sees data from these train and validation sets at train time. Specifically, we use the validation set to perform cross validation and prevent overfitting. Next, we collect the classification precision, recall, and F1-score of each anomaly detection system with this test data. Second, in order to test Rapid under domain shift, we leverage two other IoT datasets [33], [34] and two other DDoS attack traces [35], [36]. Because the additional IoT datasets contain no DDoS attack traffic, we use them to observe the negative effect each system has on benign traffic. Namely, we can count the number of false



(a) Confidence in Smart Home 1. **(b)** Confidence in Smart Home 2. **(c)** Confidence in Smart Hospital.

(d) Confidence under DARPA DDoS Attack. **(e)** Confidence under CAIDA DDoS Attack. **(f)** Confidence under Booter DDoS Attack.

Fig. 6: Cumulative distribution of classification confidences in various environments.

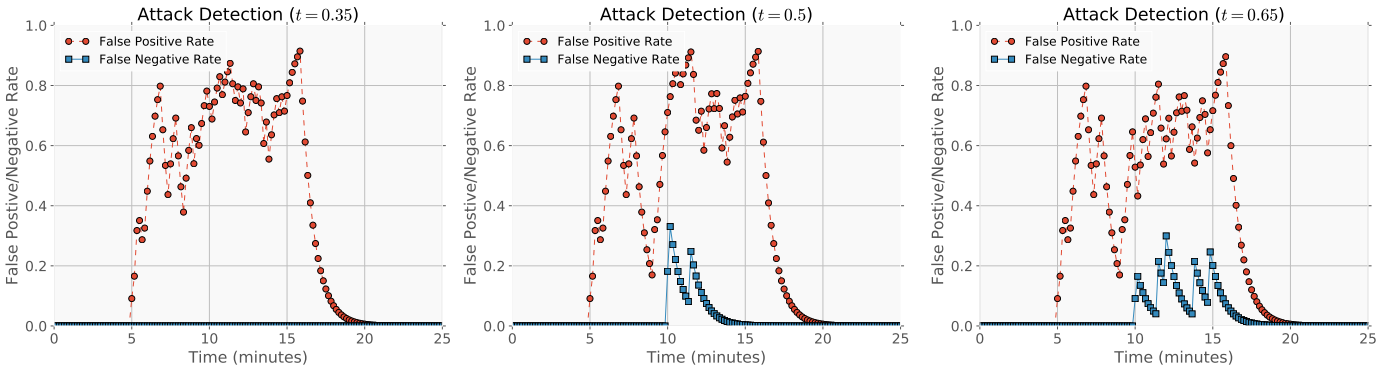
positives and calculate the sensitivity of each system. Similarly, because the additional DDoS attack traces contain no benign traffic, we use them observe how well each system can prevent DDoS attacks. Namely, we can count the number of false negatives and calculate the specificity of each system.

In line with traditional anomaly detection studies, we can see in Fig. 3 that each detection system can achieve decent performance in the network in which they train. In fact, we see that Rapid slightly out performs each system. However, Fig. 4 and Fig. 5 suggest Rapid significantly outperforms the other systems under domain shift. While Fig. 4a and Fig. 5a detail the performance gains seen in Fig. 3, Fig. 4b and Fig. 4c show that Rapid offers superior robustness to domain shift. Namely, they show Rapid limits negative effects on benign traffic with significantly better sensitivity and less false positives. While Fig. 5b and Fig. 5c suggest that Rapid's performance gains on benign traffic may come at the cost of detecting new attacks, they also show that when Rapid deploys with our novel active learning technique, Rapid can more effectively detect new DDoS attacks than any other system. In fact, our novel active learning technique limits false positives and false negatives practically to zero—even under domain shift.

C. Model Calibration and Reliability

To further discuss the deployability of each system, we next observe the *model calibration* of each system. Under proper model calibration, a model should exhibit a smooth and diverse set of confidences, but maintain a majority of confident classifications (i.e., near 0 or 100). Ideally, a system should exhibit prediction probabilities (i.e., confidences) that form a shape similar to a sigmoid curve reflected on the $y = x$ curve that plateaus on the horizontal ratio of benign to malicious traffic in the test set and points upward again only after the classification threshold. Proper model calibration limits the computational expense of attack mitigation [7].

In Fig. 6, we show the cumulative distribution function (CDF) of the confidence of each system in each environment. The horizontal dotted line in each graph shows the ratio of benign to malicious traffic in the test set, and the vertical line represents the classification threshold. Therefore, an ideal CDF will display confidences that level near the horizontal dotted line, but turn upward again after the vertical dotted line. We can



(a) Detection threshold set to 0.35. **(b)** Detection threshold set to 0.5. **(c)** Detection threshold set to 0.65.

Fig. 7: The false positive and false negative rates of our attack detection component.

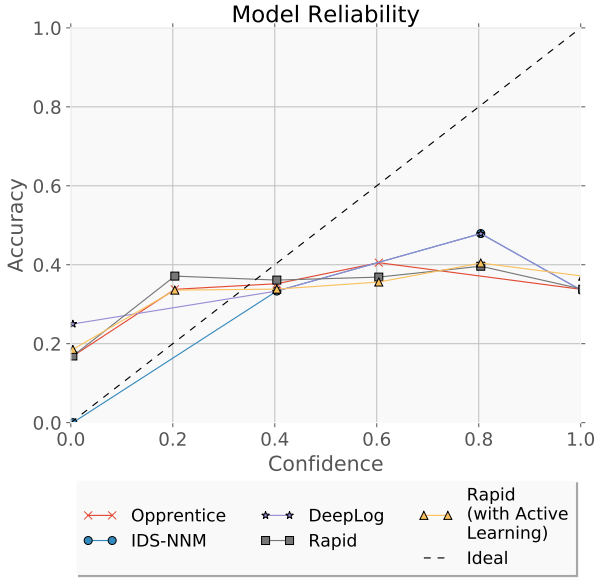


Fig. 8: Model Reliability for each defense system.

see that the two other neural network-based techniques—IDS-NNM and DeepLog—showcase poor calibration and frequently exhibit confidences very close to the decision threshold which reinforces previous studies that suggest that neural networks often exhibit poor calibration [38]. Meanwhile, while we deployed the exact same set of calibration techniques on Rapid as we did for IDS-NNM and DeepLog (i.e., temperature scaling), Rapid achieves a more acceptable calibration which suggests another benefit of our ensembled architecture with ARIMA models. Moreover, while Opprentice shows decent calibration, we see that Rapid maintains higher confidence across benign traffic in Fig. 6a, Fig. 6b, and Fig. 6c. However, without active learning, Rapid struggles to confidently detect two of the three unseen DDoS attacks in Fig. 6e and Fig. 6f. Because Rapid is so strongly confident of the opposite classification, Rapid could manually adapt to these attacks, but we would violate our assumption of no manual tuning in new environments. Fortunately, this is exactly the motivation of our novel active learning techniques. In fact, our active learning technique successfully detects all three of the unseen DDoS attacks with high confidence and proper model calibration.

Similarly, we observe a closely related metric to model calibration: the *model reliability* of each system. Under proper model reliability, a model should exhibit a correlation between accuracy and confidence. For example, ideally, 75% of classifications with confidence 0.75 should truly be attacks. Model reliability is notoriously difficult to achieve [38], and in Fig. 8, we see similar results in that each model struggles to achieve an ideal model reliability. While Rapid mostly classifies reliably for benign traffic, our evaluation suggests that each model struggles to reliably classify DDoS traffic which we pose as an area of future interest.

D. Attack Detection

Lastly, we observe the attack detection trade offs for Rapid. Because a network operator should use attack detection mainly to improve system efficiency, we aim to investigate the effect of the detection threshold for attack detection (i.e., a decrease of the classification threshold, t, should result in less false negatives but more false positives). In particular, in the Smart Home 1 network, we examine the false positive and false negative rates over time of our Rapid system adapted for attack detection. We can see in Fig. 7 that Rapid successfully adjusts for various operator desires. Specifically, in Fig. 7a, when the detection threshold is set to 0.35, Rapid has no false negatives in attack detection. With no false negatives, such a setting will have no affect on the overall classification accuracy of Rapid, but will reduce system execution. Whereas, in Fig. 7b, when the detection threshold is set to 0.5, we see a network operator can achieve a more balanced mix of false positives and false negatives, and in Fig. 7c, when the detection threshold is set to 0.65, we see Rapid can reduce false positives further at the cost of false negatives. Thus, when desired, a network operator can smoothly adjust Rapid to favor system efficiency over detection latency.

V. Conclusion

In order to identify IoT-enabled DDoS attacks that attempt to leave a policed IoT network, we presented a novel anomaly detection system, Rapid, specifically designed for real-world IoT deployment. Rapid achieves state of the art accuracy,

and further, due to our novel and automatic active learning techniques, our detection system requires little to no manual system maintenance for IoT network operators—even under domain shift. Moreover, Rapid supports diagnostic insight into classifications and operates in a real-time and online fashion. Through simulation of real-world IoT traffic traces (both malicious and benign), we showed that Rapid initially deploys with higher accuracy and confidence compared to past defenses, and further, our active learning techniques allow Rapid to adapt to a new network domain and achieve state of the art classification accuracy within a reasonable period of time.

REFERENCES

[1] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, *et al.*, "Understanding the Mirai botnet," in *Proceedings of the 26th USENIX Security Symposium*, pp. 1093–1110, USENIX, 2017.

[2] S. Herwig, K. Harvey, G. Hughey, R. Roberts, and D. Levin, "Measurement and analysis of Hajime, a peer-to-peer IoT botnet," in *Proceedings of the 2019 Network and Distributed System Security Symposium*, Internet Society, 2019.

[3] A. Wang, W. Chang, S. Chen, and A. Mohaisen, "Delving into Internet DDoS attacks by botnets: Characterization and analysis," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2843–2855, 2018.

[4] M. Lyu, D. Sherratt, A. Sivanathan, H. H. Gharakheili, A. Radford, and V. Sivaraman, "Quantifying the reflective DDoS attack capability of household IoT devices," in *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pp. 46–51, ACM, 2017.

[5] M. S. Kang, S. B. Lee, and V. D. Gligor, "The crossfire attack," in *2013 IEEE Symposium on Security and Privacy*, pp. 127–141, IEEE, 2013.

[6] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *2010 IEEE Symposium on Security and Privacy*, pp. 305–316, IEEE, 2010.

[7] S. Mergendahl, D. Sisodia, J. Li, and H. Cam, "FR-WARD: Fast retransmit as a wary but ample response to distributed denial-of-service attacks from the Internet of Things," in *Proceedings of the 27th International Conference on Computer Communication and Networks*, pp. 1–9, IEEE, 2018.

[8] D. Liu, Y. Zhao, H. Xu, Y. Sun, D. Pei, J. Luo, X. Jing, and M. Feng, "Opprentice: Towards practical and automatic anomaly detection through machine learning," in *Proceedings of the 2015 Internet Measurement Conference*, pp. 211–224, ACM, 2015.

[9] K. Azizzadenesheli, A. Liu, F. Yang, and A. Anandkumar, "Regularized learning for domain adaptation under label shifts," in *Proceedings of the 7th International Conference on Learning Representations*, 2019.

[10] C. Feng, T. Li, and D. Chana, "Multi-level anomaly detection in industrial control systems via package signatures and LSTM networks," in *Proceedings of the 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 261–272, IEEE, 2017.

[11] K. Xie, X. Li, X. Wang, J. Cao, G. Xie, J. Wen, D. Zhang, and Z. Qin, "On-line anomaly detection with high accuracy," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1222–1235, 2018.

[12] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," *ACM SIGMETRICS Performance Evaluation Review*, vol. 32, no. 1, pp. 61–72, 2004.

[13] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 219–230, 2004.

[14] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement*, pp. 71–82, 2002.

[15] K. Xie, X. Li, X. Wang, G. Xie, J. Wen, J. Cao, and D. Zhang, "Fast tensor factorization for accurate Internet anomaly detection," *IEEE/ACM Transactions on Networking*, vol. 25, no. 6, pp. 3794–3807, 2017.

[16] I. Nevat, D. M. Divakaran, S. G. Nagarajan, P. Zhang, L. Su, L. L. Ko, and V. L. Thing, "Anomaly detection and attribution in networks with temporally correlated traffic," *IEEE/ACM Transactions on Networking*, vol. 26, no. 1, pp. 131–144, 2017.

[17] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, "Bayesian event classification for intrusion detection," in *Proceedings of the 19th Annual Computer Security Applications Conference*, pp. 14–23, IEEE, 2003.

[18] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1285–1298, ACM, 2017.

[19] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *Proceedings of the 2018 Network and Distributed System Security Symposium*, Internet Society, 2018.

[20] A. Beaugnon, P. Chifflier, and F. Bach, "Ilab: An interactive labelling strategy for intrusion detection," in *International Symposium on Research in Attacks, Intrusions, and Defenses*, pp. 120–140, Springer, 2017.

[21] A. Hamza, H. H. Gharakheili, T. A. Benson, and V. Sivaraman, "Detecting volumetric attacks on IoT devices via SDN-based monitoring of MUD activity," in *Proceedings of the 2019 ACM Symposium on SDN Research*, pp. 36–48, 2019.

[22] N. Moustafa, B. Turnbull, and K.-K. R. Choo, "An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of Internet of Things," *IEEE Internet of Things Journal*, 2018.

[23] B. Sun, F. Yu, K. Wu, Y. Xiao, and V. C. Leung, "Enhancing security using mobility-based anomaly detection in cellular mobile networks," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 4, pp. 1385–1396, 2006.

[24] O. Linda, T. Vollmer, and M. Manic, "Neural network based intrusion detection system for critical infrastructures," in *2009 International Joint Conference on Neural Networks*, pp. 1827–1834, IEEE, 2009.

[25] P. Phaal, S. Panchen, and N. McKee, "InMon corporation's sFlow: A method for monitoring traffic in switched and routed networks," tech. rep., 2001.

[26] J. Mirkovic and P. Reiher, "D-WARD: A source-end defense against flooding denial-of-service attacks," in *IEEE Transactions on Dependable and Secure Computing*, vol. 2, pp. 216–232, 2005.

[27] T. M. Gil and M. Poletto, "MULTOPS: A data-structure for bandwidth attack detection," in *Proceedings of the 10th USENIX Security Symposium*, vol. 10, USENIX, 2001.

[28] S. Abdelsayed, D. Glimsholt, C. Leckie, S. Ryan, and S. Shami, "An efficient filter for denial-of-service bandwidth attacks," in *Proceedings of the Global Telecommunications Conference*, vol. 3, pp. 1353–1357, 2003.

[29] H. Wang, D. Zhang, and K. G. Shin, "Detecting SYN flooding attacks," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 1530–1539, IEEE, 2002.

[30] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: Forecasting and control*. John Wiley & Sons, 2015.

[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[32] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, 2018.

[33] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Classifying IoT devices in smart environments using network traffic characteristics," *IEEE Transactions on Mobile Computing*, 2018.

[34] M. G. Hospital, "Openice medical device network normal operation, data set 1." https://www.impactcybertrust.org/dataset_view?idDataset=1253, 2019.

[35] CAIDA, "Ucsd network telescope traffic samples." www.impactcybertrust.org, 2007.

[36] J. Santanna, R. van Rijswijk-Deij, R. Hofstede, A. Sperotto, M. Wierbosch, L. Zambenedetti Granville, and A. Pras, "Booters - an analysis of ddos-as-a-service attacks," in *IFIP/IEEE International Symposium on Integrated Network Management*, pp. 243–251, 2015.

[37] U. of Southern California-Information Sciences Institute, "Darpa 2009 ddos attack." https://www.impactcybertrust.org/dataset_view?idDataset=742, 2009.

[38] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning*, pp. 1321–1330, JMLR, 2017.