Botnet Attack Detection at the IoT Edge Based on Sparse Representation

Christos Tzagkarakis*†, Nikolaos Petroulakis*, Sotiris Ioannidis*
*Institute of Computer Science, Foundation for Research and Technology - Hellas (FORTH)
†Department of Computer Science, University of Crete, Heraklion, Greece
{tzagarak,npetro,sotiris}@ics.forth.gr

***Abstract*—Internet-of-Things (IoT) aims at interconnecting thousands or millions of smart objects/devices in a seamless way by sensing, processing and analyzing huge amount of data obtained from heterogeneous IoT devices. This rapid development of IoT-oriented infrastructures comes at the cost of increased security threats through IoT-based botnet attacks. In this work, we present an IoT botnet attack detection method based on a sparsity representation framework using a reconstruction error thresholding rule for identifying malicious network traffic at the IoT edge coming from compromised IoT devices. The botnet attack detection is performed based on small-sized benign IoT network traffic data, and thus we have no prior knowledge about malicious IoT traffic data. We present our results on a real IoT-based network dataset and show the efficacy of our proposed technique against a reconstruction error-based autoencoder approach.**

***Index Terms*—IoT edge, botnet attack detection, sparse representation, reconstruction error threshold, small-sized data**

I. Introduction

The technology of IoT has emerged during the last years [1] [2] [3] [4] as a milestone in advancing the concept of Internet networking towards connecting data, users and "things" (in the rest of the paper they are dubbed as IoT devices) in a seamless fashion. IoT technology is based on three pillars: highly heterogeneous *IoT data* are captured through a *gateway* and are immediately accessible to a wide range of applications via a secure *networking infrastructure*. The type of IoT applications span from smart homes [5] [6], smart cities [7] and wearables [8] [9] to energy management [10], predictive maintenance [11] [12], automotive driving [13], etc. However, the rapidly growing use and realization of IoT-based technology comes at the cost of resolving significant business and technical impediments as reflected in dynamicity, scalability, heterogeneity and end-to-end security/privacy [14] [15] [16].

More specific, a dynamically adaptive behaviour is followed at the IoT infrastructure, at the IoT applications and at the IoT devices, and thus it is important to promote a (semi)-automatic behaviour within all IoT layers. This gives rise to the pursuit of high scalability properties from the network layers as well as from the IoT infrastructure. In addition, enhanced heterogeneous behaviour as a result of the extensive use and interconnection of a large volume of diverse IoT devices should be addressed through the concept of efficient semantic interoperability within IoT applications and platforms. End-to-end security is also a very crucial issue since IoT devices, IoT applications and their enabling platforms could be vulnerable to various attacks.

978-1-7281-2171-0/19/$31.00 © 2019 IEEE

In the current work, the focus is given on the efficient, robust and fast detection of botnet attacks at the IoT edge. More specific, over the last few years, great research effort has been carried on IoT security and IoT network intrusion detection. As stated in [17] [18], the intrusion detection systems constitute one of the most important core components of IoT systems, and thus novel appropriate technologies should be introduced in heterogeneous environments to ensure security and privacy.

Enhanced effort is needed to tackle the integration of communication technologies and IoT in a secure middleware, able to cope with the defined protection constraints as well as with the IoT security in mobile devices. These challenges should be considered both in centralized and distributed cases, and as a result new security strategies have to be carefully designed [19] [20]. In addition, the authors in [21] provide a technical analysis and taxonomy of Distributed Denial of Service (DDoS) attacks in IoT ecosystems indicating the vulnerability of IoT devices in terms of conscription for building huge botnet armies towards performing IoT-based DDoS attacks. The study in [22] presents various challenges and opportunities related with IoT and cloud anomaly detection providing a description of the prominent features and application fields of IoT and cloud in terms of security and privacy risks.

A defence architecture based on software-defined networking (SDN) is studied in [23] for detecting and mitigating IoT DDoS attacks under a Mirai botnet assumption based on a scanning phase (and traffic) of bots to identify compromised IoT nodes. The authors in [24] propose an IoT honeypot and sandbox framework for attracting and analyzing Telnet attacks against a wide range of IoT devices running on various CPU architectures.

A game theoretic approach for lightweight anomaly detection in IoT networks is described in [25], where the concept of Nash equilibrium is adopted to determine the equilibrium state allowing the intrusion detection system to activate the anomaly detection procedure in order to detect a new attack pattern. A real-time hybrid IoT intrusion detection method based on MapReduce architecture is introduced in [26] consisting of an anomaly-based and a specification-based intrusion detection

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:46:15 UTC from IEEE Xplore. Restrictions apply.

module for detecting sinkhole and selective-forwarding IoT attacks. A self-adapting, knowledge-driven intrusion detection system for IoT able to detect attacks in real time across IoT systems running different communication protocols is proposed in [27]. The system autonomously collects information about the features of the monitored network and entities, and leverages such knowledge to dynamically configure the most effective set of detection techniques.

In [28], the authors design, implement, and evaluate a novel IoT intrusion detection system, focusing on the detection of routing attacks such as spoofed or altered information, sinkhole, and selective-forwarding. Besides, a deep autoencoder botnet attack detection method is described in [29], where a novel network-based anomaly detection method is proposed based on extracting IoT network's behavioral snapshots and adopting deep autoencoders to detect anomalous network traffic emanating from compromised IoT devices.

Contributions. We introduce a diagnosis mechanism for instant IoT botnet attack detection, with the ultimate goal of minimizing the attack's impact by immediate isolation of compromised IoT devices located at the IoT edge. As a result of the limited computational capabilities which govern the edge IoT devices, we are strongly interested in providing an algorithmic procedure which uses as small as possible amount of training and testing data towards implementing an accurate IoT botnet attack detector. Here, we assume that there is no prior knowledge of malicious IoT network traffic data during the training procedure. The novelty of the current paper is twofold. Firstly, a reconstruction error thresholding rule based on a sparse representation framework is employed for IoT botnet attack detection assuming that only a very limited amount of both training and testing data is used to deal with low computational constraints as well as with fast reaction. Secondly, a greedy sparse recovery algorithm, dubbed as orthogonal matching pursuit [30], is adopted since it involves only two hyper-parameters tuning, i.e., the thresholding constant and the sparsity level.

The rest of the paper is organized as follows: Section II briefly overviews the dataset used for the algorithmic evaluation and the corresponding feature extraction process, while Section III describes the proposed sparse representation framework along with a description of the reconstruction error thresholding rule. An experimental evaluation is provided in Section IV, where a comparative study is performed between the proposed technique and a typical autoencoder. Finally, Section V summarizes the main conclusions and gives directions for future work.

II. Dataset and feature extraction

As it is mentioned in the introductory section, we are interested in proposing an IoT botnet attack detector at the IoT edge, and thus the use of real-data is of paramount importance. Here, we used the N-BaIoT dataset[1] which corresponds to real traffic data gathered from nine commercial IoT devices. For the sake of completeness a short overview of the dataset is provided next (a detailed description can be found in [29]).

The N-BaIoT dataset contains the features extracted from the raw IoT network traffic data. More specific, whenever a packet is received, a behavioral snapshot of the protocols and hosts that transmitted each packet is computed. Each snapshot corresponds to the packet's contextual information as reflected in a set of statistical features, i.e., the arrival of each packet invokes the extraction of 23 statistical features from five time windows (100ms, 500ms, 1.5sec, 10sec and 1min), and then five 23-dimensional vectors from each window are concatenated into a single 115-dimensional vector (in the rest of the text we will refer to the 115-dimensional vector as *instance*).

For the performance evaluation, we used malicious instances obtained during a BASHLITE botnet attack. Specifically, we used the instances based on three BASHLITE attack types: (I) COMBO: sending spam data and opening a connection to a specified IP address and port, (II) Junk: sending spam data, and (III) Scan: scanning the network for vulnerable devices. The interested reader is referred to [29] for more details on the feature extraction process. For the sake of clarity, it is important to notice that the uploaded N-BaIoT dataset[1] includes a different amount of benign instances (see Table I) as compared to the dataset description in [29]. As a result, during the performance evaluation we used the benign instances that correspond to eight IoT devices as shown in the third column of Table I.

TABLE I
COMMERCIAL IOT DEVICES USED TO CAPTURE THE BENIGN INSTANCES. THE THIRD COLUMN CONTAINS THE ACTUAL NUMBER OF UPLOADED BENIGN INSTANCES[1].

Device model	Number of benign instances mentioned in [29]	Number of uploaded benign instances
Danmini	49,548	40,395
Ennio	39,100	34,692
Ecobee	13,113	13,111
Philips B120N/10	175,240	160,137
Provision PT-737E	62,154	55,169
Provision PT-838	98,514	91,555
SimpleHome XCS7-1002-WHT	46,585	42,784
SimpleHome XCS7-1003-WHT	19,528	–
Samsung SNH 1011 N	52,150	46,817

III. Sparse representation for IoT botnet attack detection

A. Proposed sparse representation framework

In this section, we provide a description of the sparse representation framework for IoT botnet attack detection using a small amount of training and testing instances. Let S be the total number of IoT devices located at the IoT edge. Then, a matrix $\mathbf{V}_i$ can be constructed for each IoT device based on the benign instances extracted from the i-th IoT device as follows

$$\mathbf{V}_i = [\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, \ldots, \mathbf{v}_{i,n_i}] \in \mathbb{R}^{d \times n_i},\ i = 1, \ldots, S\,, \qquad (1)$$

where $\mathbf{v}_{i,j} \in \mathbb{R}^{d \times 1}$ denotes the j-th d-dimensional instance of the i-th IoT device, and n_i is the number of benign training

[1] http://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:46:15 UTC from IEEE Xplore. Restrictions apply.

instances obtained from the i-th IoT device. The total number of benign training instances is $N = n_1 + \cdots + n_S$. Let us rewrite each vector $\mathbf{v}_{i,j}$ in a columnized form

$$\mathbf{v}_{i,j} = [\mathbf{v}^T_{(i,j)_1}, \ldots, \mathbf{v}^T_{(i,j)_W}]^T \in \mathbb{R}^{d\times 1}, \quad (2)$$

where $\mathbf{v}_{(i,j)_w} \in \mathbb{R}^{d_w \times 1}$ for $w = 1, \ldots, W$, $d = d_1 + \cdots + d_W$ and W denotes the number of windows used to compute the statistical features as described in Section II. According to the N-BaIoT dataset description, $W = 5$ and $d_w = 23$ for $w = 1, \ldots, W$, and thus the dimension d_w of each subvector $\mathbf{v}_{(i,j)_w}$ is constant and equal to 23 $\forall\, w = 1, \ldots, W$ (i.e., $d = d_1 + \cdots + d_W = d_1 + \cdots + d_5 = 23 + \cdots + 23 = 5 \cdot 23 = 115$).

In IoT botnet attack detection, the ultimate goal is to detect whether the IoT network traffic data corresponds to benign or malicious behaviour given an observed instance $\mathbf{y} \in \mathbb{R}^{d\times 1}$. Let us consider that $\mathbf{y}$ is an instance that corresponds to the i-th IoT device. We are interested in deducing if $\mathbf{y}$ is benign, emitted from a "healthy" IoT device, or not. The instance $\mathbf{y}$ can be written as a linear combination of the benign training instances associated with the i-th IoT device as follows

$$\mathbf{y} = c_{i,1}\mathbf{v}_{i,1} + c_{i,2}\mathbf{v}_{i,2} + \cdots + c_{i,n_i}\mathbf{v}_{i,n_i} = \mathbf{V}_i\, \mathbf{c}_i\,, \quad (3)$$

where $\mathbf{c}_i = \{c_{i,j}\}_{j=1}^{n_i}$ is the vector containing the representation coefficients of $\mathbf{y}$ in terms of the columns of $\mathbf{V}_i$.

The overall data matrix $\mathbf{V}$ contains the instances corresponding to the benign instances extracted from all IoT devices and it is defined as the concatenation of all the benign data matrices $\mathbf{V}_i$, $\forall\, i = 1, \ldots, S$,

$$\begin{aligned}\mathbf{V} &= [\mathbf{v}_{1,1}, \ldots, \mathbf{v}_{1,n_1}, \mathbf{v}_{2,1}, \ldots, \mathbf{v}_{2,n_2}, \ldots, \mathbf{v}_{S,1}, \ldots, \mathbf{v}_{S,n_S}] \\ &= [\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_S] \in \mathbb{R}^{d\times N}\ . \end{aligned} \quad (4)$$

By combining (3) and (4), $\mathbf{y}$ can be sparsely expressed in terms of the overall benign training data matrix $\mathbf{V}$, namely, $\mathbf{y} = \mathbf{V}\mathbf{c}$, where

$$\mathbf{c} = [0,\, \ldots,\, 0,\, c_{i,1},\, c_{i,2},\, \ldots,\, c_{i,n_i},\, 0,\, \ldots,\, 0]^T \in \mathbb{R}^{N\times 1} \quad (5)$$

denotes the coefficients vector, hereafter called the *sparse code*, whose elements are all zero except for those associated with the i-th IoT device.

Given the overall data matrix $\mathbf{V}$ and the observed instance $\mathbf{y}$, the following optimization problem can be solved through the orthogonal matching pursuit (OMP) [30] algorithm in order to obtain an estimate of the sparse code $\mathbf{c}$

$$\hat{\mathbf{c}} = \arg\min_{\mathbf{c}} \|\mathbf{y} - \mathbf{V}\mathbf{c}\|_2\,,\ \text{s.t.}\ \ \|\mathbf{c}\|_0 \leq \tau, \quad (6)$$

where $\|\cdot\|_2$ denotes the ℓ_2 norm, $\|\cdot\|_0$ is the ℓ_0 (pseudo)norm, which is defined as the number of non-zero elements of a given vector and τ denotes the sparsity level of the estimated sparse code $\hat{\mathbf{c}}$. Algorithm 1 summarizes the steps followed for estimating the sparse code $\mathbf{c}$ given $\mathbf{y}$ and $\mathbf{V}$. OMP is an iterative, lightweight algorithmic process, where during each iteration it selects a column of $\mathbf{V}$ to include in the current support set Λ_k (contains the indices of the selected columns) by maximizing the inner product between the columns of $\mathbf{V}$ and the current residual $\mathbf{r}_{k-1}$. Once the new column has been added to the support set, it solves a least squares problem to fully minimize the error on the current support set. As a result, the residual becomes orthogonal to the columns of $\mathbf{V}$ corresponding to the current support set. Line 2 in Algorithm 1 indicates the stopping criterion: the iterative process stops when a sparsity level τ is reached or the residual's ℓ_2 norm is below a given constant ϵ. In the current work, we consider τ as a hyper-parameter, while ϵ is fixed during the experimental evaluation.

Algorithm 1 Orthogonal matching pursuit (OMP) used for sparse code estimation.

Input: $\mathbf{y}$, $\mathbf{V}$, sparsity level τ
Output: estimated sparse code $\hat{\mathbf{c}}$

1: **Initialization:** $k = 1$, $\mathbf{r}_{k-1} = \mathbf{y}$, $\Lambda_{k-1} = \emptyset$
2: **while** $k \leq \tau$ **or** $\|\mathbf{r}_{k-1}\|_2 \leq \epsilon$ **do**
3: $\quad i^* = \arg\max_{1\leq i\leq N} |\mathbf{v}_i^T \mathbf{r}_{k-1}| / \|\mathbf{v}_i\|_2$
4: $\quad \Lambda_k = \Lambda_{k-1} \cup i^*$
5: $\quad \mathbf{c}_{\Lambda_k} = \mathbf{V}^{\dagger}_{\Lambda_k}\mathbf{y}$
6: $\quad \mathbf{r}_k = \mathbf{y} - \mathbf{V}_{\Lambda_k}\mathbf{c}_{\Lambda_k}$
7: $\quad k = k + 1$

The fundamental assumption is that if the observed instance $\mathbf{y}$ corresponds to a benign traffic behaviour, then we expect the reconstruction error $\|\mathbf{y} - \mathbf{V}\hat{\mathbf{c}}\|_2$ to be small since the indices of the non-zero entries in $\hat{\mathbf{c}}$ will correspond to those columns of $\mathbf{V}$ associated with the i-th IoT device. On the contrary, we expect a high reconstruction error $\|\mathbf{y} - \mathbf{V}\hat{\mathbf{c}}\|_2$ when $\mathbf{y}$ corresponds to an unseen (malicious) traffic behaviour since the estimated sparse code $\hat{\mathbf{c}}$ cannot be sparsely expressed in terms of $\mathbf{V}$ because the malicious IoT traffic information is not included in the overall matrix $\mathbf{V}$. As a result, the IoT botnet attack detection thresholding rule can be written as

$$\text{detection}(\mathbf{y}) = \begin{cases} \text{benign } \mathbf{y}, & \text{if } \|\mathbf{y} - \mathbf{V}\hat{\mathbf{c}}\|_2 < \theta \\ \text{malicious } \mathbf{y}, & \text{if } \|\mathbf{y} - \mathbf{V}\hat{\mathbf{c}}\|_2 \geq \theta \end{cases}, \quad (7)$$

where θ is the decision threshold. The decision threshold can be estimated offline given the overall data matrix $\mathbf{V}$ containing only benign instances.

B. Decision threshold estimation and tuning of the hyper-parameters

Algorithm 2 summarizes the main steps towards finding the best combination of the hyper-parameters τ, θ given $\mathbf{V}$. In the current work, we adopt the concept of *proxy outliers* [31] to compensate for lacking malicious instances during the hyper-parameters tuning. We assume that if the sparse codes are computed only on benign instances, some of the reconstruction errors might attain large values. As a result, choosing the maximum reconstruction error as the threshold θ could lead in accepting most of the malicious instances as benign.

The concept of quartiles is adopted to remove a certain amount of proxy outliers (attaining large reconstruction error values) present in the benign instances. More specifically, the sparse codes of all the benign training instances are computed first, and then the corresponding reconstruction errors are

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:46:15 UTC from IEEE Xplore. Restrictions apply.

estimated. Given the computed reconstruction errors, the lower quartile (Q_1), the upper quartile (Q_3) and the inter-quartile range ($IQR = Q_3 - Q_1$) is estimated, and thus an instance $\mathbf{y}$ that belongs to the benign training instances set is qualified as a proxy outlier if

$$\text{recon. error}(\mathbf{y}) < Q_1 - \rho \cdot IQR \ \texttt{or} \ \text{recon. error}(\mathbf{y}) > Q_3 + \rho \cdot IQR, \quad (8)$$

where ρ is the rejection rate reflecting the percentage of the benign training instances which fall within the non-extreme limits. Based on (8), the extreme values of the reconstruction error that represents spurious training instances can be removed, and thus the maximum of the remaining reconstruction errors is selected as the threshold θ. Besides, the best ρ value can be found through cross-validation (see Algorithm 2).

C. Majority voting for IoT botnet attack detection

As explicitly mentioned in the introductory section, the main goal of the proposed approach is the efficient and fast IoT botnet attack detection. Towards this direction, we examine a real-life scenario using *only one test instance* $\mathbf{y} \in \mathbb{R}^{115\times 1}$ in order to detect the IoT network traffic behaviour as fast as possible in a reliable manner.

Let us consider that $\mathbf{y}$ can be decomposed into five subvectors of the form $\mathbf{y}^1, \ldots, \mathbf{y}^5$ with each subvector $\mathbf{y}^w \in \mathbb{R}^{23\times 1}$ reflecting the statistical features from five time windows, 100ms ($w = 1$), 500ms ($w = 2$), 1.5sec ($w = 3$), 10sec ($w = 4$) and 1min ($w = 5$), respectively (see Section II). The optimization problem (6) can be solved for each subvector $\mathbf{y}^w$ for $w = 1, \ldots, 5$ as follows

$$\hat{\mathbf{c}}^w = \arg\min_{\mathbf{c}^w} \|\mathbf{y}^w - \mathbf{V}^w\mathbf{c}^w\|_2 \ , \ \text{s.t.} \ \|\mathbf{c}^w\|_0 \leq \tau^w, \quad (9)$$

where $\mathbf{V}^w \in \mathbb{R}^{23\times N}$ corresponds to the benign training instances of the w-th time window, and thus we end up with a set of five sparse codes $\hat{\mathbf{c}}^1, \ldots, \hat{\mathbf{c}}^5$. Next, five reconstruction errors of the form $\|\mathbf{y}^w - \mathbf{V}^w\hat{\mathbf{c}}^w\|_2$ ($w = 1, \ldots, 5$) are computed leading to five decision functions as expressed in (7). The final decision about the existence or not of an IoT botnet attack detection is provided via a majority voting scheme. It is obvious that a different decision threshold θ_w is estimated given $\mathbf{V}^w$ according to the process described in Algorithm 2 (there are inner loops of the form "**for** w = 1 to 5 **do**" within the loops in Line 3, Line 15 and Line 30). Due to space limitation, an extended description of Algorithm 2 including a window-based hyper-parameters tuning pseudocode, will be provided in an upcoming publication.

IV. Experimental evaluation

In this section, the IoT botnet attack detection performance of the proposed sparse representation (SR) method based on majority voting is compared against a single hidden layer autoencoder (AE) [32], where the N-BaIoT dataset (see Table I) was used during the evaluation process. For each IoT device we randomly select 100, 300 and 500 benign instances from the first half of each dataset to estimate the decision threshold and perform the tuning of the hyper-parameters following a 3-fold ($CV = 3$) cross-validation process, where

Algorithm 2 Decision threshold estimation and tuning of the hyper-parameters for IoT botnet attack detection

Input: matrix $\mathbf{V}$ containing the benign training instances, set $\mathcal{T}$ of all possible τ values, set $\mathcal{P}$ of all possible ρ values, number of cross validation folds CV

Output: estimated decision threshold θ, (best) tuned sparsity level $\tau \in \mathcal{T}$, (best) tuned rejection rate $\rho \in \mathcal{P}$

1: **for** $\rho \in \mathcal{P}$ **do** // loop over all possible values of ρ
2: | **for** $\tau \in \mathcal{T}$ **do** // loop over all possible values of τ
3: | | **for** $i = 1$ to N **do** // loop over all columns of $\mathbf{V}$
4: | | | $\mathcal{I} = \{1, \ldots, N\}$
5: | | | $\mathbf{b} = \mathbf{V}(:, i)$ // extract the i-th column of $\mathbf{V}$
6: | | | $\mathcal{I}(i) = []$ // remove the i-th index
7: | | | // matrix after removing the i-th column:
8: | | | $\tilde{\mathbf{V}} = \mathbf{V}(:, \mathcal{I})$
9: | | | $\hat{\mathbf{c}}_i = \arg\min_{\mathbf{c}_i} \|\mathbf{b} - \tilde{\mathbf{V}}\mathbf{c}_i\|_2 \ , \ \text{s.t.} \ \|\mathbf{c}_i\|_0 \leq \tau$
10: | | | $\hat{\mathbf{b}} = \tilde{\mathbf{V}}\hat{\mathbf{c}}_i$ // reconstructed version of $\mathbf{b}$
11: | | | $\mathbf{e}(i) = \left((1/d)\sum_{l=1}^{d}\left(\mathbf{b}(l) - \hat{\mathbf{b}}(l)\right)\right)^{1/2}$ // RMSE error
12: | | Compute the lower quartile Q_1 of the reconstruction errors $\mathbf{e}$.
13: | | Compute the upper quartile Q_3 of the reconstruction errors $\mathbf{e}$.
14: | | Compute the inter-quartile range $IQR = Q_3 - Q_1$.
15: | | **for** $i = 1$ to N **do** // loop over all elements in $\mathbf{e}$
16: | | | **if** $\mathbf{e}(i) < Q_1 - \rho \cdot IQR$ **or** $\mathbf{e}(i) > Q_3 + \rho \cdot IQR$ **then**
17: | | | | // i-th training instance in $\mathbf{V}$ is
| | | | | // considered as proxy outlier:
18: | | | | $\mathbf{V}_m(:, i) = \mathbf{V}(:, i)$
19: | | | **else**
20: | | | | // i-th training instance in $\mathbf{V}$ is
| | | | | // considered as benign:
21: | | | | $\mathbf{V}_n(:, i) = \mathbf{V}(:, i)$
22: | | | | // $\mathbf{e}_n$ contains the recon. errors of
| | | | | // instances considered as benign:
23: | | | | $\mathbf{e}_n(i) = \mathbf{e}(i)$
24: | | For the current set of values (ρ, τ) compute the decision threshold θ as the maximum value of $\mathbf{e}_n$.
25: | | **for** $f = 1$ to CV **do** // loop over cross val. folds
26: | | | Current "test" cross validation fold f used to build the benign instances matrix for testing purposes: $\mathbf{D}_f$.
27: | | | Current "training" cross valid. folds $\{1, \ldots, CV\} \setminus \{f\}$ used to build the benign instances matrix for training purposes: $\mathbf{D}_r$.
28: | | | $\mathbf{D}_t = [\mathbf{D}_f \ \mathbf{V}_m]$ // overall testing data
29: | | | // loop over all columns of matrix $\mathbf{D}_t$:
30: | | | **for** $j = 1$ to columns($\mathbf{D}_t$) **do**
31: | | | | $\mathbf{y} = \mathbf{D}_t(:, j)$ // select j-th test instance
32: | | | | $\hat{\mathbf{s}}_j = \arg\min_{\mathbf{s}_j} \|\mathbf{y} - \mathbf{D}_r\mathbf{s}_j\|_2 \ , \ \text{s.t.} \ \|\mathbf{s}_j\|_0 \leq \tau$
33: | | | | $\hat{\mathbf{y}} = \mathbf{D}_r\hat{\mathbf{s}}_j$ // estimated version of $\mathbf{y}$
34: | | | | // RMSE error between $\mathbf{y}$ and its
| | | | | // estimated version $\hat{\mathbf{y}}$:
35: | | | | $\mathbf{r}(j) = \left((1/d)\sum_{l=1}^{d}\left(\mathbf{y}(l) - \hat{\mathbf{y}}(l)\right)\right)^{1/2}$
36: | | | | **if** $\mathbf{r}(j) < Q_1 - \rho \cdot IQR$ **or** $\mathbf{r}(j) > Q_3 + \rho \cdot IQR$ **then**
37: | | | | | // instance $\mathbf{y}$ is considered benign:
38: | | | | | $\boldsymbol{\alpha}(j) = 1$
39: | | | | **else**
40: | | | | | // instance $\mathbf{y}$ is considered malicious:
41: | | | | | $\boldsymbol{\alpha}(j) = 0$
42: | | | // compute the geometric mean based on $\boldsymbol{\alpha}$:
43: | | | $\mathbf{g}(f) = \sqrt{TPR \cdot TNR}$
44: | | // compute the average geometric mean:
45: | | $\bar{\mathbf{g}} = (1/CV)\sum_{l=1}^{CV}\mathbf{g}(l)$

46: Select as the (best) tuned set (ρ, τ), denoted as $(\tilde{\rho}, \tilde{\tau})$, the one corresponding to the maximum value of the geometric mean $\bar{\mathbf{g}}$.

47: The estimated decision threshold is denoted as $\tilde{\theta}$ and corresponds to the best combination of hyper-parameters $(\tilde{\rho}, \tilde{\tau})$.

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:46:15 UTC from IEEE Xplore. Restrictions apply.

TABLE II
CONFUSION MATRIX CORRESPONDING TO THE EVALUATION RESULTS ON IOT BOTNET ATTACK DETECTION.

Actual \ Detected	Malicious	Benign
Malicious	True Positive (TP)	False Negative (FN)
Benign	False Positive (FP)	True Negative (TN)

τ is varied from $\mathcal{T} = \{5, 10, 15, 20, 30\}$, ρ is varied from $\mathcal{P} = \{0.01, 0.5, 1, 2, 3\}$, and ϵ is fixed and equal to 0.001. For the AE hyper-parameters tuning we followed a similar strategy as the one analyzed in Algorithm 2 (based on an AE reconstruction error-oriented decision threshold estimation and hyper-parameters tuning), where the number of epochs is fixed and equal to 50, while the number of nodes in the hidden layer is varied from $\{20, 30, 40, 50, 60\}$. As a result, both SR and AE have one hyper-parameter, the sparsity level and the number of nodes, respectively. Here, an off-the-shelf AE implementation was used[2], where `KerneScale` parameter was set to `auto` and `Standardize` to `true`, while the rest of the parameters were kept to default values.

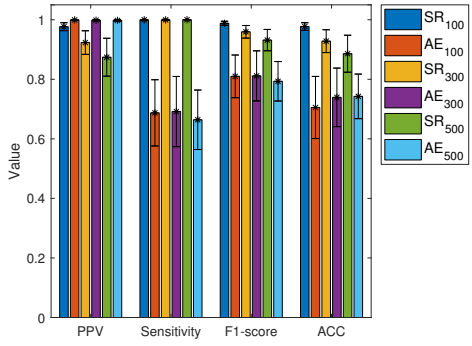

Fig. 1. Performance evaluation results for the COMBO botnet attack.

The evaluation results on IoT botnet attack detection are reported in the form of a confusion matrix as shown in Table II, where TP indicates the quantity of malicious instances correctly detected, TN shows the quantity of benign instances correctly detected, FN indicates the quantity of malicious instances incorrectly detected, and FP denotes the quantity of benign instances incorrectly detected. Here, we calculated the following metrics based on the confusion matrix in order to assess the performance of the proposed framework: (I) Positive Predictive Value (PPV) which indicates the proportion of correctly detected malicious instances in the total instances detected as malicious, (II) Sensitivity (detection rate) which shows the proportion of correctly detected malicious instances in the total number of actual malicious instances, (III) F1-score corresponding to the harmonic mean of PPV and sensitivity, (IV) Accuracy (ACC) denoting the fraction of correctly detected instances in total detected instances.

To evaluate the performance of the two methods, we performed five Monte Carlo runs. During each Monte Carlo run we followed a leave-one-out-device-out cross validation (LOOCV) strategy, where benign instances from $S-1$ IoT devices were used for tuning and threshold estimation, while the current (under testing) IoT device's benign and malicious instances were used for testing/evaluation purposes. This procedure was repeated S times and the total average performance metrics over all IoT devices and all Monte Carlo runs are reported. This evaluation is IoT device independent and shows the generalization capabilities as the IoT device which is being tested is not included in the tuning procedure.

[2]http://www.mathworks.com/help/nnet/ref/trainautoencoder.html

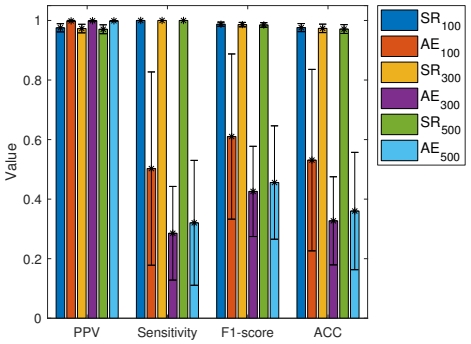

Fig. 2. Performance evaluation results for the Junk botnet attack.

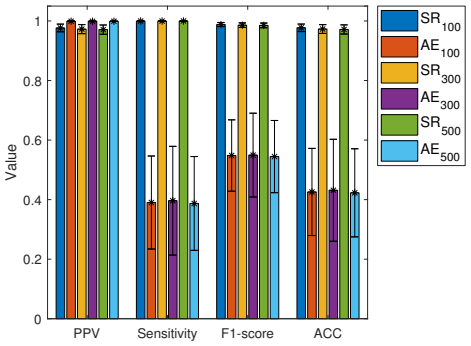

Fig. 3. Performance evaluation results for the Scan botnet attack.

During the evaluation process, we used 100, 300 and 500 left-out benign instances (see LOOCV description in the previous paragraph), respectively, for testing as well as 200 malicious instances randomly selected from each IoT device's COMBO malicious dataset (1600 malicious testing instances in total). In the case of Junk and Scan botnet attack we used 200, 600 and 1000 randomly selected instances from each IoT device's malicious Junk and Scan dataset, respectively (1600, 4800 and 8000 malicious testing instances in total during each evaluation scenario). It is important to notice that we used the malicious instances obtained from the eight IoT devices used during the tuning process (see Table I). Figure 1 shows the results corresponding to the COMBO botnet attack, Figure 2 depicts the performance in the case of Junk botnet attack and Figure 3 corresponds to the Scan botnet attack results. In all figures, the subscripts in the legend names indicate the number of benign instances per IoT device used during the hyper-parameters tuning and the decision threshold estimation process. The vertical black lines indicate the error bars since each experimental scenario is performed five (Monte Carlo runs) by $S = 8$ (total number of IoT devices) times.

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:46:15 UTC from IEEE Xplore. Restrictions apply.

It is obvious that the proposed SR method achieves superior performance in light of Sensitivity, F1-score and ACC, while the AE technique achieves slightly better results in terms of PPV. That means that SR is robust in accurately detecting both malicious and normal behaviour in the IoT network (the Sensitivity, F1-score and ACC error bars corresponding to AE are wider as compared to the SR method's error bars). Besides, the time complexity between SR and AE is comparable and low (due to space limitation, a more thorough computation cost investigation will be provided in a future publication), and thus SR can be applied for accurate and fast IoT botnet attack detection.

V. Conclusions

In this paper, we proposed a method for fast IoT botnet attack detection based on a very small amount of benign training instances and using one instance during detection. The proposed approach is based on a sparse representation framework, where the decision threshold is estimated using only benign training instances. The sparse representation method is compared with a single hidden layer autoencoder. It was shown through an experimental evaluation that the proposed method performs better in terms of F1-score, detection rate and accuracy than the autoencoder. As a future work, we intend to examine the proposed approach using more IoT botnet attack datasets as well as performing extensive comparisons with additional IoT botnet attack detection methods.

Acknowledgment

This work has received funding from the European Union Horizon's 2020 research and innovation programme under grant agreement No. 780315 (SEMIoTICS).

References

[1] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 414–454, First 2014.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, Fourthquarter 2015.

[3] L. D. Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, Nov 2014.

[4] J. A. Stankovic, "Research directions for the internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 3–9, Feb 2014.

[5] B. L. R. Stojkoska and K. V. Trivodaliev, "A review of internet of things for smart home: Challenges and solutions," *Journal of Cleaner Production*, vol. 140, pp. 1454 – 1464, 2017.

[6] S. D. T. Kelly, N. K. Suryadevara, and S. C. Mukhopadhyay, "Towards the implementation of IoT for environmental condition monitoring in homes," *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3846–3853, Oct 2013.

[7] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, Feb 2014.

[8] J. Wei, "How wearables intersect with the cloud and the internet of things : Considerations for the developers of wearables." *IEEE Consumer Electronics Magazine*, vol. 3, no. 3, pp. 53–56, July 2014.

[9] S. Amendola, R. Lodato, S. Manzari, C. Occhiuzzi, and G. Marrocco, "RFID technology for IoT-based personal healthcare in smart spaces," *IEEE Internet of Things Journal*, vol. 1, no. 2, pp. 144–152, April 2014.

[10] F. Shrouf and G. Miragliotta, "Energy management based on internet of things: practices and framework for adoption in production management," *Journal of Cleaner Production*, vol. 100, pp. 235 – 246, 2015.

[11] F. Civerchia, S. Bocchino, C. Salvadori, E. Rossi, L. Maggiani, and M. Petracca, "Industrial internet of things monitoring solution for advanced predictive maintenance applications," *Journal of Industrial Information Integration*, vol. 7, pp. 4 – 12, 2017.

[12] D. Kwon, M. R. Hodkiewicz, J. Fan, T. Shibutani, and M. G. Pecht, "IoT-based prognostics and systems health management for industrial applications," *IEEE Access*, vol. 4, pp. 3659–3670, 2016.

[13] W. He, G. Yan, and L. D. Xu, "Developing vehicular data cloud services in the IoT environment," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1587–1595, May 2014.

[14] A. Botta, W. de Donato, V. Persico, and A. Pescap, "Integration of cloud computing and internet of things: A survey," *Future Generation Computer Systems*, vol. 56, pp. 684 – 700, 2016.

[15] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for internet of things: A survey," *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 70–95, Feb 2016.

[16] I. Lee and K. Lee, "The internet of things (IoT): Applications, investments, and challenges for enterprises," *Business Horizons*, vol. 58, no. 4, pp. 431 – 440, 2015.

[17] B. B. Zarpelo, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in internet of things," *Journal of Network and Computer Applications*, vol. 84, pp. 25 – 37, 2017.

[18] S. Sicari, A. Rizzardi, L. Grieco, and A. Coen-Porisini, "Security, privacy and trust in internet of things: The road ahead," *Computer Networks*, vol. 76, pp. 146 – 164, 2015.

[19] R. H. Weber, "Internet of things new security and privacy challenges," *Computer Law & Security Review*, vol. 26, no. 1, pp. 23 – 30, 2010.

[20] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266 – 2279, 2013.

[21] R. Hallman, J. Bryan, G. Palavicini, J. Divita, and J. Romero-Mariona, "IoDDoS The internet of distributed denial of service attacks - A case study of the mirai malware and IoT-based botnets," *Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security*, vol. 1: IoTBDS, no. 47-58, 2017.

[22] I. Butun, B. Kantarci, and M. Erol-Kantarci, "Anomaly detection and privacy preservation in cloud-centric internet of things," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, June 2015, pp. 2610–2615.

[23] M. Ozcelik, N. Chalabianloo, and G. Gur, "Software-defined edge defense against IoT-based DDoS," in *2017 IEEE International Conference on Computer and Information Technology (CIT)*, Aug 2017, pp. 308–313.

[24] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, "IoTPOT: A novel honeypot for revealing current IoT threats," *Journal of Information Processing*, vol. 24, no. 3, pp. 522–533, 2016.

[25] H. Sedjelmaci, S. M. Senouci, and M. Al-Bahri, "A lightweight anomaly detection technique for low-resource IoT devices: A game-theoretic methodology," in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[26] H. Bostani and M. Sheikhan, "Hybrid of anomaly-based and specification-based IDS for internet of things using unsupervised OPF based on MapReduce approach," *Computer Communications*, vol. 98, pp. 52 – 71, 2017.

[27] D. Midi, A. Rullo, A. Mudgerikar, and E. Bertino, "Kalis – A system for knowledge-driven adaptable intrusion detection for the internet of things," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, June 2017, pp. 656–666.

[28] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: Real-time intrusion detection in the internet of things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661 – 2674, 2013.

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

[30] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. on Information Theory*, vol. 53(12), pp. 4655–4666, December 2007.

[31] S. S. Khan, M. E. Karg, D. Kulic, and J. Hoey, "Detecting falls with X-factor hidden Markov models," *Applied Soft Computing*, vol. 55, pp. 168 – 177, 2017.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:46:15 UTC from IEEE Xplore. Restrictions apply.