



Anomaly Detection for Power Grid Based on Network Flow

Lizong Zhang¹, Xiang Shen¹, Fengming Zhang¹, Minghui Ren¹,
and Bo Li²(✉)

¹ State Grid Shaoxing Power Supply Company, Zhejiang, China
z1z951@163.com, shx8022@163.com, zhangfm712@163.com,
17226656@163.com

² School of Computer Science and Engineering, Beihang University,
Beijing, China
libo@act.buaa.edu.cn

Abstract. As an important part of the national infrastructure, the power grid is facing more and more network security threats in the process of turning from traditional relative closure to informationization and networking. Therefore, it is necessary to develop effective anomaly detection methods to resist various threats. However, the current methods mostly use each packet in the network as the detection object, ignore the overall timing pattern of the network, cannot detect some advanced behavior attacks. In this paper, we introduce the concept of network flow, which consists of the same end-to-end network packets, besides the network flow fragmentation divides the network flow into pieces at regular intervals. We also propose a network flow anomaly detection method based on density clustering, which uses bidirectional flow statistics as features. The experimental result demonstrate that the methodology has excellent detection effect on large-scale malicious traffic and injection attacks.

Keywords: Power grid · Anomaly detection · Network flow · Density cluster

1 Introduction

Over the past few decades, the emergence and development of the next generation smart grid has enabled the grid to improve in all aspects of power generation, transmission, distribution and consumption. It improves the energy efficiency, maximizes the utility, reduces the cost and controls the emission [1]. However, the smart grid breaks the previous physically isolated power network, causing the adversary has more possible access points and intrude the entire network. Recently, cyber-attacks against the power grid are gradually increasing, among which the Ukrainian grid event is the most concerned. On December 23, 2015, the Ukrainian Kyivoblenergo's seven 110 kV and 23 35 kV substations were disconnected for three hours. After investigation, this event was considered to be due to a foreign attacker remotely controlled the SCADA distribution management system [2]. The incident exposed the Ukrainian grid with many security vulnerabilities, for example, VPNs into the ICS from the business network lacks the two-factor authentication, and there is no abnormal detection or

active defense devices. This reflects the necessity of anomaly detection and defense in the smart power grid.

As an important industrial control system, the power grid is a key national infrastructure. In the development process from traditional closed architecture to informationization and networking open architecture, numerous vulnerabilities are gradually exposed. To protect national security, many countries are investing in industrial security (including power system security) and make certain progress [3]. Existed research work includes behavior-based, rule-based detection methods such as [4, 5], which is mainly based on the deep analysis of the industrial control protocol, extracts the key field information (combined of expert experience partially), and detects through the access control list such as whitelist and blacklist. In addition, the model-based anomaly behavior detection uses the system model parameters and features to establish mathematical models to detect intrusion anomaly, such as AAKR, CUSUM, ARIMA [6–8], besides the formal models can be constructed for anomaly detection by extracting programmable logical controller code logic [9, 10]. Lastly, the machine learning methods are also studied widely in industrial control anomaly detection, mainly divided into supervised learning, unsupervised learning and semi-supervised learning, including K-means, fuzzy C-means, support vector machine, intelligent Markov Chains, etc. [11–13].

The above methods have made progress in the detection, but the data elements are in units of packets, lacking the correlation feature of time dimension. The state machine model only pays attention to the order of the packets, but ignores the time limit. In this paper, we consider the same end-to-end network packet collection as a network flow, divide the network flow to form fragments at regular intervals, and extract time-related statistical features from the fragment. We use unsupervised density clustering as a machine learning method. In the training phase, we use the normal samples as input, the normal network flow segmentation will be clustered into non-quantitative clusters according to the distance between features. The boundary of each cluster constitutes the anomaly detection model. In the detection phase, when the network flow feature is too far away from any clusters, it is considered abnormal.

In summary, the contributions of this paper as follows:

- (1) We introduce the concept of network flow and fragmentation, which allows us to extract and exploit time dimension features.
- (2) We propose a network flow anomaly detection method based on density clustering, which is intuitively effective to differentiate between network flows and construct the model of network flows.

The rest of the paper is organized as follows: Sect. 2 provides the background on network flow and density clustering. Section 3 presents the algorithm's framework and its entire training phase and execution phase. Section 4 presents experimental results in terms of model's construction and detection performance. Finally, in Sect. 5 we present our conclusion.

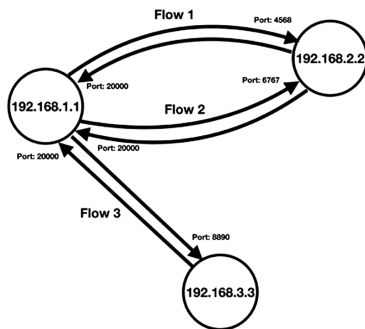


Fig. 1. Network flow diagram

2 Background

2.1 Network Flow and Fragment

A network flow is a collection of end-to-end bidirectional packets in the network. In the most common TCP/IP architecture, the collection of packets in each TCP session constitutes a network flow. However, it should be noticed that different connections between two nodes are not the same network flow (different ports in TCP), because the tasks they perform are not same necessarily, the communication mode may be different. Besides, during extracting the network flow features, since the two-way communication mode is not same necessarily, each direction packets will be feature-extracted separately to avoid mixing. Finally, it should be noted that the TCP session ends after the four-way handshake, or after the timeout expires, the flows ends, regardless of subsequent network flows. Network flow diagram shown in Fig. 1.

However, it is unacceptable to perform statistical feature extraction until the end of each network session to, especially the network session in the industrial control system (including the power grid) will last for a long time. In order to enable feature extraction and detection, we introduce the concept of network flow fragmentation, which divide the network flow at regular intervals, the packets in every interval.

2.2 Density Clustering

Density clustering can cluster irregular shapes without requiring to predetermine the number of clusters, and discrete point noise data can be processed better. Density clustering assumes that the clustering structure can be determined by the tightness of the sample distribution. In general, density clustering determines the connectivity between samples by their density, and the clusters are expanded continuously by connectable samples to obtain the final clustering results.

DBSCAN is a representative algorithm of density clustering, which use $(\epsilon, \text{MinPts})$, two neighborhood parameters to describe the degree of distribution between cluster samples, the following will define several basic concepts by the data set $S = \{x_1, x_2, \dots, x_n\}$:

ε Neighborhood. For each $x_i \in S$, its neighborhood ε represents a set of samples in which distance from x_i is no greater than ε , denoted as $N_\varepsilon(x_i) = \{x_j \in S | \text{dist}(x_i, x_j) \leq \varepsilon\}$;

Core Object. If the ε neighborhood of x_i contains at least MinPts samples, i.e. $|N_\varepsilon(x_i)| \geq \text{MinPts}$, then x_i can be called the core object;

Density Directly Reachable: If x_i is a core object, and x_j is in the ε neighborhood of x_i , then x_j can be density reached directly by x_i ;

Density Reachable: If there is a sample sequence $\theta_1, \theta_2, \dots, \theta_n$, $\theta_1 = x_i$, $\theta_n = x_j$, and any θ_{k+1} in the sequence can be density reached directly by θ_k , the x_j is determined by x_i density reachable;

Density Connection: If x_k for x_i, x_j , such that x_i, x_j , are all density reachable by x_k , then x_i is connected to x_j density [14].

Through the above definitions, a cluster can be described as a sample set consisting of a density-reachable, maximum density connected. In DBSCAN clustering algorithm, it is necessary to determine all the core objects in a given data set by neighborhood parameters (ε , MinPts), find out the objects whose density is reachable for each core object, and form cluster clusters. If an object is reachable by more than one core object simultaneously, these core objects and their clusters are merged into the same cluster. When all core objects are traversed and expanded, the density clustering result is obtained, besides the objects which do not belong to any cluster are considered noise.

3 Anomaly Detection Model

3.1 Overview

The industrial control system network mainly performs timing data acquisition and scheduling control, which has high periodicity and certainty. During normal operation of the system, the statistical information of each network flow should be in a stable interval, and the characteristic values of each network flow should be gathered in a tight cluster, it is natural to build a model using density clustering to detect abnormal.

3.2 Feature Extraction and Preprocessing

The network flow we propose is bidirectional, and the nodes at both ends are divided into a server and a client according to whether or not the service is provided. Therefore, the network flow has two directions: toClient and toServer. In addition, the two-way network flow has different communication modes, so it is necessary to distinguish and extract features separately.

In the two directions of network flow, we extract the number of bytes of each packet and the interval time of the packets, and calculate the mean and variance respectively to better fit the distribution. In addition, we also extract the information entropy of the byte, which indicates the distribution of ASCII code per byte in each packet, which is the implicit mode of network transmission. In Table 1, the specific characteristics are shown.

Table 1. Network flow features

| | Feature | Quantity | Meaning |
|--|---------------|----------|--|
| Statistical features (toServer and toClient) | Bytes.avg | 2 | Mean of packet bytes |
| | Bytes.std | 2 | Standard deviation of packet bytes |
| | Intervals.avg | 2 | Mean of packet intervals |
| | Intervals.std | 2 | Standard deviation of packet intervals |
| | Be | 2 | Byte information entropy |

In this paper, the characteristics of network stream extraction are all numeric types, which are comparable float numbers. However, the difference of the original data magnitude of each feature is large, so that the weights of the features of the final clustering result are different, so it is necessary to normalize the features. In this paper, the Sigmoid curve is selected as the function of the normalization operation, the mean and standard deviation parameters are introduced in the basis of the original formula, so that the normalized feature data is concentrated near 0.5 and has certain linearity.

$$\sqrt{y} = \frac{1}{1 + e^{-\frac{y-avg}{std}}}$$

3.3 Training Phase

The process of constructing the cluster anomaly detection model is roughly as follows:

Step 1: Data acquisition, obtaining sample data for constructing the model through the network flow engine (samples composed of 11 statistical class features);

Step 2: Data preprocessing, calculating the mean avg of each feature, the standard deviation std, and normalizing each feature of each sample using a Sigmoid function;

Step 3: Density clustering, given empirical neighborhood parameters (ϵ , MinPts), performing density clustering on the normalized sample data to obtain cluster clusters to which each sample belongs;

Step 4: According to the clustering result, the range value is obtained by calculating the maximum value of each feature in each cluster. As the cluster boundary of the normal model, the network flow anomaly detection model of the normal mode is constructed.

3.4 Detection Phase

After the cluster anomaly detection model is constructed, it will consist of the following steps in the model detection phase:

Step 1: Data acquisition, through the network flow engine, whenever the network flow reaches the fragmentation time, obtain the statistical sample data;

Step 2: Data preprocessing, the obtained sample data, the mean value avg calculated by the model construction step, and the standard deviation std parameter are normalized by the Sigmoid function;

Step 3: Anomaly detection, comparing the normalized sample data with the cluster boundary one by one, and obtaining the degree of abnormality of the sample from each cluster feature boundary (Euclidean distance, if a certain feature of the sample is at Within the cluster boundary, the feature and its boundary distance are recorded as 0). If there is a cluster such that the sample abnormality is less than the abnormal threshold, the detection result is normal; if the abnormality of all cluster samples is greater than the abnormal threshold, the detection result is abnormal.

4 Experiment

4.1 Experiment Procedure

This experiment is completed on the laboratory industrial control system simulation platform, which mainly includes the following steps:

- (1) Start the HMI and PLC to enter the normal communication mode, and observe the data of the HMI interface is in a normal state;
- (2) Perform network stream data buffering, calculate the mean value and variance of each feature after the buffer amount is satisfied, to perform data normalization;
- (3) Training the pre-processed samples to construct a network flow clustering model;
- (4) The simulated HMI suffers from DDos attack (distributed denial of service attack, through a large number of legitimate requests, preempting network resources, causing the server network to smash), a large number of network flows should occur in a short time, but the network flow characteristics and industrial control under normal mode. There is a huge difference in network traffic, and it is observed whether the network flow clustering model can detect anomalies;
- (5) Simulate a large number of malicious injection attacks on PLC. A large number of injection attacks will bring about changes in network flow statistics, observe whether the network flow clustering model can detect abnormalities.

4.2 Experiment Results

The experimental results and analysis are as follows:

- (1) In a highly periodic industrial control network, network flow features are densely distributed at multiple points, so the clustering target is to gather multiple small clusters to precisely represent the feature range. In the algorithm implementation, the parameters in the density cluster are set to $\varepsilon = 0.05$ and $\text{MinPts} = 10$, respectively. Moreover, the algorithm will normalize the 12 features of the selected network flows, represent the clustering results in the form of maximum and minimum values of each cluster, as shown in Fig. 2. Analysis of network flow data in normal mode, including a total of 1 Modbus connection, 3 S7 connections, so the clustering of 4 clusters is a

reasonable result. Figure 3 shows the results of network flow clustering mapping each cluster's features to the radar graph.

```

Network flow cluster --- -bash --- 99x24
ABB-iMac:Network flow cluster apple$ python dbscan.py flow_5_5 ddos

cluster result(4 clusters):
max: [0.414, 0.692, 0.696, 0.439, 0.437, 0.39, 0.621, 0.659, 0.451, 0.439]
min: [0.411, 0.689, 0.695, 0.438, 0.437, 0.388, 0.57, 0.618, 0.448, 0.438]
avg: [0.412, 0.69, 0.696, 0.439, 0.437, 0.389, 0.595, 0.639, 0.45, 0.439]

max: [0.56, 0.485, 0.481, 0.425, 0.425, 0.604, 0.591, 0.549, 0.419, 0.424]
min: [0.553, 0.484, 0.48, 0.425, 0.425, 0.597, 0.589, 0.548, 0.419, 0.424]
avg: [0.556, 0.484, 0.48, 0.425, 0.425, 0.601, 0.59, 0.548, 0.419, 0.424]

max: [0.679, 0.45, 0.431, 0.423, 0.423, 0.65, 0.48, 0.439, 0.419, 0.423]
min: [0.665, 0.446, 0.429, 0.423, 0.423, 0.639, 0.474, 0.438, 0.419, 0.423]
avg: [0.672, 0.448, 0.43, 0.423, 0.423, 0.645, 0.477, 0.439, 0.419, 0.423]

max: [0.362, 0.366, 0.383, 0.811, 0.719, 0.37, 0.336, 0.363, 0.811, 0.719]
min: [0.36, 0.366, 0.383, 0.702, 0.704, 0.369, 0.334, 0.363, 0.701, 0.704]
avg: [0.361, 0.366, 0.383, 0.756, 0.712, 0.369, 0.335, 0.363, 0.756, 0.712]

```

Fig. 2. Cluster result

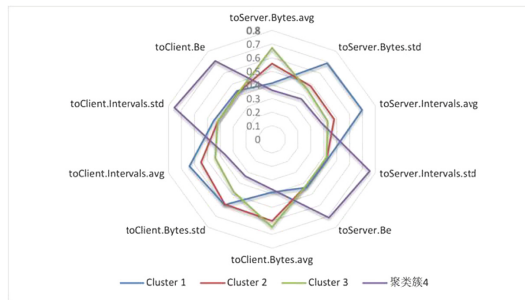


Fig. 3. Cluster result radar graph

(2) The simulated HMI is subjected to a DDoS attack, and a large number of spurious S7 request packets with a destination port of 102 are sent, so that a large number of network flows occur in a short period of time, each network flow contains only a small number of packets. There is a notable difference between the industrial control network flow and the normal mode. Figure 4 shows the abnormal results detected by the network flow clustering model, its feature deviation exceeds the threshold.

```

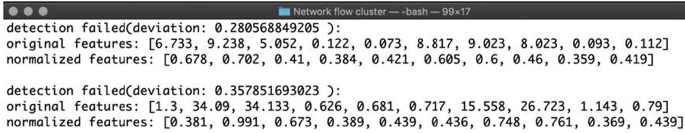
Network flow cluster --- -bash --- 99x24
detection failed(deviation: 0.46198878296 ):
original features: [0.033, 16.0, 16.0, 0.001, 0.0, 0.017, 0.0, 0.0, 0.0, 0.0]
normalized features: [0.316, 0.871, 0.511, 0.383, 0.419, 0.422, 0.368, 0.326, 0.358, 0.416]

detection failed(deviation: 0.461987813927 ):
original features: [0.033, 16.0, 16.0, 0.002, 0.0, 0.017, 0.0, 0.0, 0.0, 0.0]
normalized features: [0.316, 0.871, 0.511, 0.383, 0.419, 0.422, 0.368, 0.326, 0.358, 0.416]

```

Fig. 4. Detection result

(3) The simulated PLC suffers from a large number of malicious injection attacks, which will generate statistical changes to the network flow. Figure 5 shows the detection result of the network flow clustering model for the injected attack network flow.



```

Network flow cluster — bash — 99x17
detection failed(deviation: 0.280568849205 ):
original features: [6.733, 9.238, 5.052, 0.122, 0.073, 8.817, 9.023, 8.023, 0.093, 0.112]
normalized features: [0.678, 0.702, 0.41, 0.384, 0.421, 0.605, 0.6, 0.46, 0.359, 0.419]

detection failed(deviation: 0.357851693023 ):
original features: [1.3, 34.09, 34.133, 0.626, 0.681, 0.717, 15.558, 26.723, 1.143, 0.79]
normalized features: [0.381, 0.991, 0.673, 0.389, 0.439, 0.436, 0.748, 0.761, 0.369, 0.439]

```

Fig. 5. Detection result

5 Conclusion

This paper proposed a novel approach to detect abnormality in the industrial control system network. Based on the density clustering method of network flow statistical information, the number of clusters is determined autonomously according to the distribution density of sample points, the cluster results are tight, and abnormal noise can be found in the training process. The experiments shown that the model has excellent detection effect on large-scale traffic attacks and injection attacks, also can perform well when the system faces unknown proprietary protocols.

References

1. Fang, X., Misra, S., Xue, G., et al.: Smart grid — the new and improved power grid: a survey. *IEEE Commun. Surv. Tutorials* **14**(4), 944–980 (2012)
2. E-ISAC, SANS ICS: Analysis of the Cyber Attack on the Ukrainian Power Grid, p4, 18 March 2016. http://www.nerc.com/pa/CI/ESISAC/Documents/E-ISAC_SANS_Ukraine_DUC_18Mar2016.pdf
3. Shang, W., An, P., Wan, M., et al.: Summary of research and development of industrial control system intrusion detection technology. *Appl. Res. Comput.* **34**(2), 328–333 (2017)
4. Khalili, A., Sami, A.: SysDetect: a systematic approach to critical state determination for Industrial intrusion detection systems using Apriori algorithm. *J. Process Control* **32**(11), 154–160 (2015)
5. Choi, S., Chang, Y., Yun, J.H., et al.: Traffic-Locality-Based Creation of Flow Whitelists for SCADA Networks (2015)
6. Yang, D., Usynin, A., Hines, J.W.: Anomaly-based intrusion detection for SCADA systems (2005)
7. Mo, Y., Chabukswar, R., Sinopoli, B.: Detecting integrity attacks on SCADA systems. *IEEE Trans. Control Syst. Technol.* **44**(1), 11239–11244 (2011)
8. Zhang, Y., Tong, W., Zhao, Y.: Improvement of CUSUM anomaly detection algorithm and application in industrial control intrusion detection system. *Metallurgical Industry Automation* (2014)
9. Guo, S., Wu, M., Wang, C.: Symbolic execution of programmable logic controller code. In: *ACM SIGSOFT Symposium on the Foundations of Software Engineering*. ACM (2017)

10. Stephen, M., et al.: A trusted safety verifier for process controller code. In: NDSS Symposium 2014, pp. 1–3, February 2014
11. Zhou, C., Huang, S., Xiong, N., et al.: Design and analysis of multimodel-based anomaly intrusion detection systems in industrial process automation. *IEEE Trans. Syst. Man Cybernet. Syst.* **45**(10), 1345–1360 (2017)
12. Ponomarev, S., Atkison, T.: Industrial control system network intrusion detection by telemetry analysis. *IEEE Trans. Dependable Secure Comput.* **13**(2), 252–260 (2016)
13. Macdermott, A., Shi, Q., Merabti, M., et al.: Intrusion detection for critical infrastructure protection. In: The 13th Post Graduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting (PGNet 2012) (2012)
14. Zhou, Z.: *Machine Learning*. Tsinghua University Press, Beijing (2016)