# Detecting Peer-to-Peer Botnets in SCADA Systems

Huan Yang*, Liang Cheng† and Mooi Choo Chuah‡
Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA 18015
Email: *huy213@lehigh.edu, †cheng@cse.lehigh.edu, ‡chuah@cse.lehigh.edu

***Abstract*—Supervisory Control and Data Acquisition (SCADA) systems monitor and control critical infrastructure such as the smart grid. As SCADA systems become increasingly interconnected and adopt more and more cyber-enabled components, the risks of cyber attacks become a major concern. Due to their decentralized organization, peer-to-peer (P2P) botnets are resilient to many existing takedown measures and can be exploited as an effective way to launch cyber attacks on SCADA systems. However, little work has been done to detect P2P botnets in SCADA systems, which carry traffic flows with characteristics significantly different from the Internet. In this paper, we design a P2P-botnet detection method for SCADA systems, leveraging built-in traffic monitoring capabilities of SCADA networking devices. The proposed method consists of two stages. In the first stage, we design a simple feature test to filter out non-P2P hosts, which significantly reduces the data volume for P2P-botnet identification. In the second stage, we jointly consider flow-based and connectivity-based features that effectively set apart bots from benign hosts. We propose to use unsupervised learning for P2P-botnet identification, which not only identifies known P2P botnets but also captures newly emerged ones. Our simulation results show that the proposed system achieves high detection rates with very few false positives. Furthermore, our evaluation shows that the proposed method can detect hosts running P2P SCADA applications that are infected by P2P bots.**

## I. Introduction

Supervisory Control and Data Acquisition (SCADA) systems have been deployed to monitor and control electrical power grids for decades. As conventional power grids around the globe evolve into smart grids, an increasing number of SCADA systems get connected to remote SCADA systems, corporate networks, and even the Internet [1], [2]. To support the ever-growing series of smart-grid applications, the integration of state-of-the-art information technologies, such as cloud computing, has also been proposed [3], [4], which will introduce more computers into SCADA systems. The increasing interconnectivity, deployment of computers, and use of commercial-off-the-shelf (COTS) computing devices [5] expose SCADA systems to a vast assortment of cyber attacks, including denial-of-service (DoS) attacks, data interception, data alteration, and false data injection. Vulnerabilities of cyber-enabled components will lead to not only cyber but also disastrous physical consequences. For instance, the Stuxnet worm infected field devices in a nuclear facility and forced centrifuge speeds outside the normal operating range [6]. Among all the current threats to SCADA systems, botnets are at the top of the list as it is an effective way to launch miscellaneous attacks and disseminate malicious software [7]. Peer-to-peer (P2P) botnets are an emerging threat to SCADA systems because they are more resilient to existing takedown measures [8]. To secure the increasingly interconnected SCADA systems with more and more cyber-enabled components, it is important to detect P2P botnets.

Unlike early botnets relying on a centralized command and control (C&C) server, a P2P botnet has a decentralized C&C infrastructure, allowing bots to exchange C&C messages in a P2P manner [7]. Several notable examples of P2P botnets, such as Sality, Kelihos, and ZeroAccess, have been alive in the wild for a long time and can be leveraged to launch cyber attacks on SCADA systems. Recently proposed detection schemes (e.g., [9], [10]) are typically designed and evaluated under Internet environment, but little work has been done for P2P-botnet detection in SCADA systems. Some existing methods require the installation of sophisticated, operating-system-specific software to monitor system calls, leading to a high deployment barrier and/or operating costs for SCADA systems. Furthermore, SCADA traffic has several distinct characteristics that are not present in the Internet traffic [11]. Both the unique characteristics of SCADA traffic and the potentially devastating impacts of cyber attacks entail the design of a P2P-botnet detection algorithm for SCADA systems.

In this paper, we design a P2P-botnet detection method for SCADA systems leveraging the traffic monitoring capabilities of the SCADA network infrastructure. Our evaluation results show that our method achieves high detection accuracy with very few false positives under different scenarios. The contributions of this work are listed as follows:

- We study the problem of P2P botnet detection for SCADA systems, which have distinct traffic characteristics and diverse architectures. We show that P2P bots (i.e., cyber-enabled components infected by a certain P2P botnet) can be identified in SCADA systems by analyzing flow statistics collected by networking devices.
- As many SCADA systems organize its sensors, actuators, and protection devices in a decentralized fashion, the corresponding traffic patterns bear certain resemblances to P2P communication. To filter out non-P2P hosts and reduce the data volume for botnet detection, we incorporate a simple feature test in the preprocessing stage of our method to identify hosts engaging in (benign or botnet) P2P communication.
- To set apart P2P bots from benign P2P hosts, the bot identification stage of our method jointly considers their flow-based and connectivity-based behaviors. We propose an unsupervised approach in this stage, which not only identifies known P2P botnets but also alerts system administrator to newly emerged ones.

- In our evaluation, we consider the scenario where some bot-infected hosts simultaneously run P2P SCADA applications and show that our method is able to separate hosts infected by different types of P2P botnets from those running benign P2P applications.

## II. Related Work

### A. SCADA System Architecture and Traffic Patterns

Many SCADA systems implement sensing, computation, and control tasks through a decentralized or distributed architecture. For example, in smart grid substations, sensors and actuators are deployed near power system equipment (e.g., transformers and circuit breakers). Data collected by sensors is transmitted through the SCADA network infrastructure to multiple SCADA hosts (e.g., protective relays). Deriving the states of the power system from the data, different SCADA hosts cooperate to implement control functionality by exchanging information with each other and sending commands to actuators. SCADA traffic patterns bear certain resemblances to P2P communication, which is also decentralized in nature.

SCADA systems exhibit traffic characteristics that are significantly different from the Internet, which may render existing detection schemes targeting Internet environment less effective. In [11], traffic characteristics of SCADA systems in two water treatment and distribution facilities are studied. Unlike Internet traffic that is strongly correlated with human activity, it is found that SCADA traffic is not self-similar and does not present obvious diurnal or nocturnal patterns. Instead, the studied SCADA systems are dominated by periodic traffic generated by sensors, resulting in a large number of flows with nearly constant throughputs. From the perspective of network traffic monitoring and analysis, a SCADA system may generate the following categories of traffic patterns to support its various tasks and services:

- *Periodic data transmission.* Sensors can be configured to transmit data to other SCADA hosts periodically and autonomously. A sensor may support multiple sampling frequencies simultaneously to facilitate tasks requiring different levels of data granularity.
- *Periodic polling.* A SCADA host (e.g., a protective relay) can periodically request data from a set of sensors it specifies. A task relying on periodic polling is activated at regular time intervals.
- *Unsolicited response.* In this mode, sensors can spontaneously initiate message transmission to report status change or event occurrence without receiving any polling command. This mode is supported by DNP3.
- *Event-triggered commands.* Using the data collected from sensors, a SCADA host can determine whether certain commands need to be sent. It is common to observe multiple command sequences on a SCADA network during the execution of event-triggered tasks.
- *P2P data transmission.* To increase system resilience in the presence of faulty or compromised devices, P2P communication is also leveraged by SCADA systems. In [12], a self-organized structured P2P overlay is deployed on top of SCADA network, taking full advantage of path redundancy and data replication that are inherent to P2P technology. In the P2P convergecast application [13], data sources and sinks can be located on various hierarchical levels of an interconnected SCADA system. These communicating peers may be geographically far apart from each other.

### B. Data Collection for Botnet Detection

The life cycle of a P2P botnet consists of multiple stages (or phases), i.e., infection stage, rally stage, waiting stage, and execution stage [14]. One of the most essential characteristics of P2P botnets is their C&C channel. In the rally stage, an infected bot performs peer discovery and joins the P2P botnet. In the waiting stage, it waits for commands from the botmaster, which are delivered in a P2P fashion. Without relying on a centralized C&C server, P2P botnets are found to be resilient to takedown measures targeting centralized C&C infrastructure [8], [15]. Many P2P-botnet detection techniques distinguish C&C communication patterns from regular ones over the Internet.

To identify a botnet through its C&C communication patterns, data needs to be collected by monitoring the behaviors of individual hosts and/or network traffic flows. In MalFlow [16], host-level data flow profiles are constructed to identify malign network domains by intercepting network-related system calls. However, such a host-based approach monitors the behaviors of processes running on individual hosts, which is not feasible for SCADA systems because the incurred overhead can affect the performance of safety-critical real-time SCADA tasks.

Another approach to data collection is to capture packet traces on individual network hosts. Packet traces are then converted into a conversation-based or flow-based data set. A conversation is uniquely identified by a two-tuple (i.e., source and destination addresses). Time series of packet sizes and packet inter-arrival timestamps of individual conversations can be extracted for feature selection [14], [10]. A network flow is identified by a unique five-tuple (i.e., source and destination IP addresses, source and destination ports, and protocol). A recently proposed detection system [17] extracts flow-based statistical features from time series of packet sizes and inter-arrival timestamps of packets. Collecting packet traces on individual SCADA hosts may also introduce significant system overhead and affect the performance of critical SCADA tasks.

A third way to collect network-level data is to leverage the traffic monitoring capabilities built into switches and routers (e.g., NetFlow records). In addition to the five-tuple that uniquely identifies a network flow, a flow record can include flow arrival timestamp, flow duration, as well as summarized information such as the number of packets and the number of bytes transmitted. Implemented in many networking devices deployed in SCADA systems, traffic monitoring allows system operators to configure how flow records should be collected. When a flow terminates or its pre-specified timer expires, the corresponding flow record is exported to traffic analysis server designated by the system administrator. As collecting flow records does not incur extra overhead on existing SCADA hosts, sensors, and actuators, a botnet detection method leveraging flow records has a low deployment barrier for SCADA systems. In PeerClean [9], NetFlow records are collected at the edge routers of a campus network and supervised multi-class SVMs are trained to label P2P bot clusters. In contrast, our approach is unsupervised and is able to differentiate newly emerged bots provided that they exhibit distinct flow-based and/or connectivity-based characteristics.

## III. System Overview

The primary objective of this work is to design a P2P-botnet detection method that can be easily integrated into existing and emerging SCADA systems. Figure 1 shows the architecture of our proposed P2P-botnet identification system.

**Data Collection and System Deployment:** Network flow records are collected by networking devices (e.g., Ethernet switches and routers) of a SCADA system. A server is set up to collect the flow records and execute our P2P-botnet detection algorithm. For a hierarchically organized SCADA system, multiple detection servers can be distributedly deployed in subsystems at different levels (e.g., substation automation systems and control center network). To minimize data transmission workload between subsystems, each detection server is responsible for identifying P2P bots within its own subsystem. A detection server can be configured to signal the system administrator on site and/or send alarms to a remote location.

**Detection Algorithm:** Our two-stage detection algorithm is deployed on the detection server. In the first stage, flow records for both non-P2P and P2P hosts are used as input. We extract the good connection ratios of individual hosts and perform a simple feature test to filter out non-P2P SCADA hosts. This step identifies P2P hosts on the SCADA system and significantly reduces the data volume for the second stage. Our feature test uses good connection ratios because SCADA hosts engaging in P2P communication generate many failed connections during peer discovery. P2P hosts identified by the first stage may include those running P2P SCADA application (i.e., benign P2P hosts), non-P2P hosts infected by P2P botnet, and hosts running both P2P bot processes and P2P SCADA applications. In the second stage, we use an unsupervised learning algorithm, i.e., affinity propagation clustering, to classify benign P2P hosts and different types of known P2P bots. By jointly considering flow-based and connectivity-based features that characterize the C&C communication patterns of botnets, our algorithm can not only set apart known P2P bots but also detect P2P bots of unknown types. Our detection method is executed at the end of each monitoring period to identify P2P bots. Various lengths of monitoring period are supported provided that P2P bots generate enough network flows for C&C communication during that period. System operator can configure the length of detection periods so that P2P bots are identified in a timely manner.

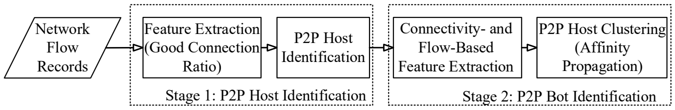

Network Flow Records
Feature Extraction (Good Connection Ratio)
P2P Host Identification
Stage 1: P2P Host Identification
Connectivity- and Flow-Based Feature Extraction
P2P Host Clustering (Affinity Propagation)
Stage 2: P2P Bot Identification

Fig. 1. System architecture overview

## IV. System Design

To design an effective method for P2P-botnet detection, we select features that differentiate P2P bots from other hosts in SCADA systems. In this section, we describe our design of the two stages of our algorithm and discuss the rationale behind their feature extraction modules.

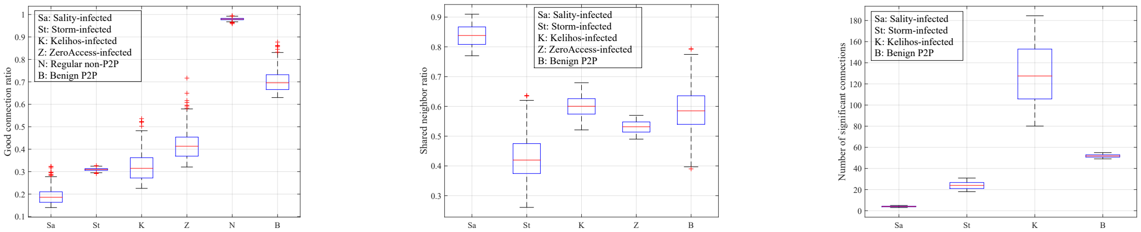
### *A. P2P Host Identification*

A SCADA system generates a huge volume of control and monitoring traffic. The majority of these flows are associated with regular non-P2P SCADA applications. Only a few tasks requiring data redundancy and resilience to sensor faults leverage P2P communication. Therefore, we first filter out hosts running only non-P2P applications. To ensure the scalability of this step, we design a simple feature test using the good connection ratio feature.

**Good Connection Ratio:** A pair of hosts share a “good” TCP connection if they complete a three-way handshake (i.e., SYN, SYN/ACK, and then ACK). They share a “good” UDP connection if UDP flow generated by one host is “responded” by the other. During monitoring period $T_i$, the good connection ratio for a host pair is defined as the ratio of the number of good connections to the total number of connections between them. As a SCADA host communicates with multiple other hosts, a set of pairwise good connection ratios is associated to it. We compute the mean and standard deviation over the set of good connection ratios of a host $j$ and denote this feature vector as $G_j^i$. Figure 2a shows the pairwise good connection ratios associated to different types of hosts in a simulated SCADA system over 24 hours (see Section V). Note that regular SCADA hosts only running non-P2P applications have significantly higher good connection ratios than P2P hosts. Hosts in SCADA systems typically assume well-defined roles in various tasks: Sensors monitor power-system equipment and send data to hosts that require the states of these equipment. These hosts respond to sensors to confirm data delivery, making connections from sensors good connections. Protective relays detect power-system events and send commands to actuators (e.g., circuit breakers), which have to respond to these commands by sending updated states of the power-system equipment being controlled. Therefore, the majority of connections between actuators and protective relays are also good ones. On the other hand, both P2P SCADA applications and P2P bots need to perform peer discovery so as to join

TABLE I
Connectivity-based statistical features

| Feature | Description |
|---|---|
| Good connection ratio | mean and standard deviation of per-host pairwise good connection ratios |
| Shared neighbor ratio | mean and standard deviation of per-host pairwise shared neighbor ratios |
| Number of significant connections | number of significant connections for each host |



(a) Good connection ratio feature

(b) Shared neighbor ratio feature

(c) Significant connections

Fig. 2. Connectivity-based features. Note that good connection ratio is used in both the P2P host identification stage and the P2P bot identification stage.

a P2P network, which results in bad connections if some of the peers are not active or in states other than peer discovery (e.g., the execution stage of P2P bots). We note that the good connection ratio feature is also used in the next stage to help differentiate P2P bots from benign P2P hosts. Both the mean and standard deviation (i.e., the $G^i_j$ feature vector) are used in the second stage.

**P2P Host Identification:** We observe that the hourly average good connection ratios associated to individual hosts can be used to identify P2P hosts. We inspect the 24-hour training data set (see Figure 2a) and empirically determine a threshold of $0.94$. The P2P host identification stage consists of a threshold-based feature test: If a host has an hourly average good connection ratio over $0.94$, we label it as a non-P2P SCADA host at the end of the monitoring period. Otherwise, it is labeled as a P2P host for further processing in the next stage. Our experiment (see Section V) shows that all the non-P2P hosts can be filtered out by this simple feature test, which significantly reduces the data volume for further analysis.

### B. P2P Bot Identification

To identify hosts infected by P2P botnet, we jointly consider two sets of features to distinguish P2P-botnet C&C traffic patterns from those generated by benign P2P hosts. The first set of features includes connectivity-based features listed in Table I. We use these feature to capture the unique characteristics of connectivity patterns of various P2P botnets.

**Good Connection Ratio:** As shown in Figure 2a, good connection ratio can be further exploited to differentiate P2P bots and benign P2P hosts. P2P SCADA applications are typically used in a small set of hosts whose tasks require data redundancy and/or resilience to interruption. After the first several rounds of peer discovery, each host will retain an up-to-date list of peers, leading to a higher good connection ratio afterwards. Using mean and standard deviation of per-host pairwise good connection ratios, we can also set apart different P2P bots. Although most P2P bots have good connection ratios lower than those of benign P2P hosts, their respective C&C communication patterns lead to distinctive characteristics of good connection ratios.

TABLE II
FLOW-BASED STATISTICAL FEATURES FOR P2P BOT IDENTIFICATION

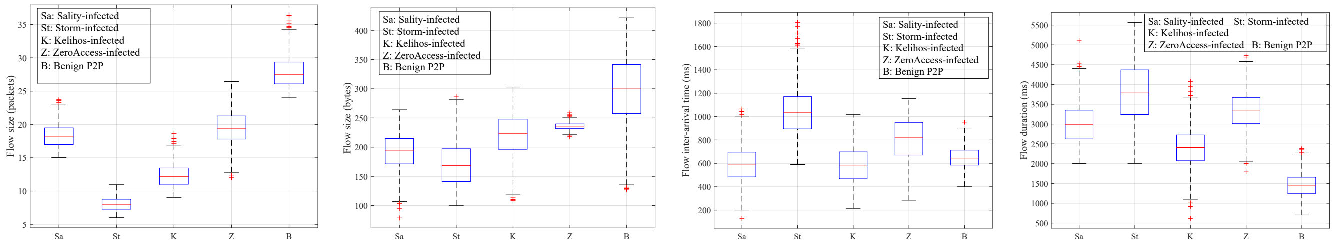
| Feature | Description |
|---|---|
| Packet-oriented flow size | mean and standard deviation of per-host flow sizes in number of packets |
| Byte-oriented flow size | mean and standard deviation of per-host flow sizes in number of bytes |
| Flow inter-arrival time | mean and standard deviation of flow inter-arrival timestamps for each host |
| Flow duration | mean and standard deviation of flow durations for each host |

**Shared Neighbor Ratio:** During monitoring period $T_i$, we find the set of P2P hosts host $j$ has contacted and denote this set by $n^i_j$. For any host $k \in n^i_j$, we find the set $n^i_k$. The pairwise shared neighbor ratio of host pair $\{j, k\}$ is defined as $n^i_{(j,k)} = \frac{||n^i_j \cap n^i_k||}{||n^i_j \cup n^i_k||}$. Mean and standard deviation of the set of pairwise shared neighbor ratios of host $j$ is used as a feature for $j$. We denote this feature vector by $N^i_j$. Figure 2b shows the pairwise shared neighbor ratios of different P2P hosts over 24 hours. This feature vector can help us effectively differentiate different types of P2P bots. In particular, Sality-infected hosts have high shared neighbor ratios because they share the same bootstrap list of peers. As all SCADA hosts must remain in operation, contents of the peer lists of Sality-infected hosts gradually evolve to become similar.

**Number of Significant Connections:** At the end of monitoring period $T_i$, we count the number of significant connections of individual P2P hosts. For host $j$, this scalar feature is denoted by $S^i_j$. Figure 2c shows the significant connection counts of different P2P hosts over $24$ hours. We observe that ZeroAccess-infected hosts have virtually no significant connections, which indicates that ZeroAccess is designed to evenly distribute its C&C workload. As shown in Figure 2c, this feature can effectively set apart Sality bots, Storm bots, Kelihos bots, and benign P2P hosts.

The second set of features includes flow-based statistical features listed in Table II. We use these features to characterize the traffic flows generated by P2P bots of different types.

**Packet-Oriented Flow Size:** At the end of monitoring period $T_i$, we compute the mean and standard deviation of flow sizes in number of packets for every individual host. Note that inbound and outbound flows are considered together. For host $j$, we denote its packet-oriented flow size feature vector as $P^i_j$. Figure 3a shows the packet-oriented flow size statistics for different types of P2P hosts over $24$ hours. Packet-based flow sizes can be used to set apart benign P2P SCADA hosts from bot-infected hosts. P2P SCADA applications transmit sensor data to other hosts exploiting the inherent path redundancy of P2P technology. The number of packets transmitted by a benign P2P host outnumbers that of a P2P bot because its P2P connections are used to transmit multiple data packets sampled at high rates: A sensor may need to collect as many as $256$



(a) Packet-oriented flow size feature (b) Byte-oriented flow size feature (c) Flow inter-arrival time feature (d) Flow duration feature

Fig. 3. Flow-based statistical features

samples per cycle (i.e., $1/60$ second) and each sample has to be sent immediately to ensure control system responsiveness. In contrast, P2P bots transmit fewer packets over established connections. These packets contain botmaster commands that need to be disseminated among bots. The differences in packet-oriented flow sizes among P2P bots can be attributed to their peer discovery strategies and command sets. Bots with a richer command set tend to have more possible states and more complicated control sequences.

**Byte-Oriented Flow Size:** At the end of monitoring period $T_i$, we compute the byte-oriented flow size feature vector $B_j^i$ for host $j$, which includes mean and standard deviation of inbound and outbound flow sizes in number of bytes. Figure 3b shows the byte-oriented flow size statistics of different P2P hosts over 24 hours. P2P SCADA applications are used to transmit sensor data packets which are normally in several hundred of bytes, whereas many C&C packets are of shorter lengths. The differences in traffic volumes of flows of various P2P bots reflect their unique characteristics of both C&C sequences and encoding schemes of commands.

**Flow Inter-Arrival Time:** In addition to flow sizes, we characterize the temporal patterns in which a P2P host communicates with its peers. For monitoring period $T_i$, we collect and sort the inbound and outbound flow arrival timestamps of host $j$, which results in a time series of flow arrival timestamps. We subtract timestamp $T_i$ from the first element of the time series and then generate a time series of flow inter-arrival timestamps. Mean and standard deviation of inter-arrival timestamps are extracted as a feature vector for host $j$, which is denoted as $I_j^i$. In a SCADA system, remote servers can collect data from sensors within the system via the remote terminal unit (RTU) [13]. Although sensors may aggregate data sampled at different time intervals before transmission, many safety-critical SCADA tasks require low latency in data delivery, limiting the amount of data accumulated. This also explains the smaller standard deviation of flow inter-arrival timestamps for benign P2P hosts in Figure 3c. In addition, Figure 3c indicates that this feature vector is not very effective in differentiating Sality-infected bots from Kelihos-infected bots because the ranges of their flow inter-arrival time almost completely overlap.

**Flow Duration:** For monitoring period $T_i$, we collect inbound and outbound flow durations for host $j$ to form a time series of flow durations. Mean and standard deviation of this time series are then computed and used as a feature vector for host $j$. We denote this feature vector by $D_j^i$. Flow durations of benign P2P hosts are relatively shorter than those of P2P bots. This can be attributed to the fact that data packets generated by a sensor can be transmitted through different P2P connections: During each cycle, a sensor collects multiple data samples, demultiplexes them over different established connections, and then tears down the connections. In contrast, flow durations of P2P bots are typically longer because these bots are deliberately designed to communicate stealthily to evade the radars of existing detection systems.

**P2P Botnet Clustering.** We apply feature normalization to all the elements in feature vector $x_j^i$: For feature element $x_j^i(l)$, we compute $\frac{x_j^i(l)-x_{l,\min}}{x_{l,\max}-x_{l,\min}}$, where $x_{l,\min}$ and $x_{l,\max}$ are the minimum and maximum feature element values among all hosts during period $T_i$, respectively. Similarities among all the pairs of P2P hosts, which are computed based on normalized features, are used by the affinity propagation algorithm to automatically find the proper number of clusters for a presented data set. Suppose that the normalized features for hosts $j$ and $k$ during monitoring period $T_i$ are denoted by vectors $\bar{x}_j^i$ and $\bar{x}_k^i$, respectively. The similarity $s^i(j,k)$ between the two host is defined as the negative Euclidean distance, i.e., $s^i(j,k) = -||\bar{x}_j^i - \bar{x}_k^i||^2$. As our bot identification algorithm is unsupervised, we expect that it can identify both known and unknown P2P bots.

## V. Evaluation

Without loss of generality, we construct two evaluation scenarios by simulating substation SCADA systems.

**Experiment I – Detecting Unknown Bots:** To demonstrate the effectiveness of our proposed method, we first evaluate whether our system can identify known and unknown bots in a SCADA system. The SCADA system simulated in OMNeT++ is depicted in Figure 4. Current sensors (CT), voltage sensors (VT), and circuit breakers (CB) are deployed near power-system equipment (e.g., transformers). Traffic patterns between protective relays and sensors can be periodic data transmission, periodic polling, or unsolicited response. Traffic patterns between protective relays and circuit breakers are

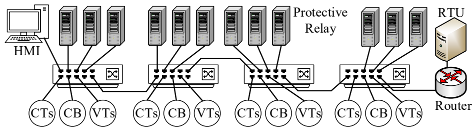


Fig. 4. A substation SCADA system simulated using OMNeT++

periodic polling and/or event-triggered commands. A remote control center communicates with the simulated substation via the RTU using multiple protocols including P2P convergecast [13]. System operators can also monitor and control the substation through the human-machine interface (HMI). There are 8 SCADA hosts generating regular non-P2P traffic. In addition, there are 10 P2P hosts where there are 2 Sality-infected hosts, 2 Kelihos-infected hosts, 2 ZeroAccess-infected hosts, 2 Storm-infected hosts and 2 P2P hosts generating benign P2P traffic. All the bots generate malicious P2P traffic while simultaneously running non-P2P SCADA applications. We generate 24-hour training data without activating the Storm bots and train the clustering algorithm. Then, we collect testing data by running the simulation for another 24 hours and activating the storm bots 8 hours after simulation starts. The second column in Table III summarizes the detection performance of our proposed method on the testing data set over 24 monitoring periods (i.e., length of $T_i$ is set to 1 hour). We note that the simple feature test employed in the first stage filters out all the 8 regular non-P2P hosts without incorrectly including any P2P hosts. Before Storm bots are activated, 4 clusters are generated by the clustering algorithm at the end of each monitoring period: 1 for benign P2P hosts, and 3 for the bots seen during training. After Storm bots are activated, a new cluster containing Storm-infected hosts is generated. This indicates that our method can be used to identify previously unseen P2P botnets and signal system administrator to take necessary measures. We further note that our method does not classify any bot as benign P2P hosts.

**Experiment II – Detecting P2P hosts infected by P2P bots:** We further consider the scenario where all the hosts run P2P SCADA applications. A SCADA system with 10 P2P SCADA hosts are simulated: There are 2 Sality-infected hosts, 2 Storm-infected hosts, 2 Kelihos-infected hosts, 2 ZeroAccess-infected hosts, and 2 benign P2P hosts communicating with remote data collection servers. After training on a 24-hour data set, we evaluate our method on a 24-hour testing set and summarize the detection performance in the third column of Table III. At the end of each monitoring period, the clustering step generates 5 clusters. Although the first stage of our algorithm does not filter out any host, the second stage is still able to achieve high detection accuracy with very few false positives and no false negatives.

## VI. Conclusion

In this work, we design an unsupervised algorithm for P2P botnet detection in SCADA systems by capturing the flow-based and connectivity-based characteristics of C&C communication patterns of bots. Our method has a low deployment barrier because it leverages traffic monitoring capabilities that have already been built into existing networking devices. High detection accuracy is achieved in our evaluation, which indicates that the proposed approach is well-suited for securing SCADA systems. As future work, we will compare our scheme with existing methods under SCADA environment. We will also evaluate the effectiveness of our scheme with new P2P bots that bypass the P2P node discovery phase, which will be a more likely scenario in SCADA environments.

TABLE III
P2P Botnet Identification Performance (A-Accuracy, P-Precision, R-Recall)

| Bot Type | Experiment I (A/P/R) | Experiment II (A/P/R) |
|---|---|---|
| Sality | 98.6%/95.6%/91.7% | 97.1%/95.6%/89.6% |
| Kelihos | 98.1%/87.0%/97.9% | 96.7%/90.0%/93.8% |
| ZeroAccess | 99.3%/94.1%/100% | 98.8%/94.1%/100% |
| Storm | 98.3%/90.1%/93.8% | 97.9%/92.2%/97.9% |

## Acknowledgment

The material is based upon work supported by the Department of Energy under Award Number DE-OE0000779.

## References

[1] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, “A Survey on Cyber Security for Smart Grid Communications,” *Commun. Surveys Tuts.*, vol. 14, no. 4, pp. 998–1010, 4 Quarter 2012.

[2] B. Zhu, A. Joseph, and S. Sastry, “A Taxonomy of Cyber Attacks on SCADA Systems,” in *Proc. of the 2011 IEEE Int. Conf. on Internet of Things, and Cyber, Physical and Social Computing (iThings/CPSCom)*, 2011, pp. 380–388.

[3] S. Bera, S. Misra, and J. J. P. C. Rodrigues, “Cloud Computing Applications for Smart Grid: A Survey,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1477–1494, 2015.

[4] A. Sajid, H. Abbas, and K. Saleem, “Cloud-Assisted IoT-Based SCADA Systems Security: A Review of the State of the Art and Future Challenges,” *IEEE Access*, vol. 4, pp. 1375–1384, 2016.

[5] D. Papp, Z. Ma, and L. Buttyan, “Embedded Systems Security: Threats, Vulnerabilities, and Attack Taxonomy,” in *Proc. of the 2015 13th Annu. Conf. on Privacy, Security and Trust*, 2015, pp. 145–152.

[6] B. Miller and D. Rowe, “A Survey of SCADA and Critical Infrastructure Incidents,” in *Proc. of the 1st Annu. Conf. on Research in Information Technology (RIIT ’12)*, 2012, pp. 51–56.

[7] S. S. Silva, R. M. Silva, R. C. Pinto, and R. M. Salles, “Botnets: A Survey,” *Computer Networks*, vol. 57, no. 2, pp. 378–403, 2013.

[8] Y. Nadji, M. Antonakakis, R. Perdisci, D. Dagon, and W. Lee, “Beheading Hydras: Performing Effective Botnet Takedowns,” in *Proc. of the 2013 ACM SIGSAC Conf. on Computer & Communications Security (CCS ’13)*, 2013, pp. 121–132.

[9] Q. Yan, Y. Zheng, T. Jiang, W. Lou, and Y. T. Hou, “PeerClean: Unveiling Peer-to-Peer Botnets through Dynamic Group Behavior Analysis,” in *Proc. of the 2015 IEEE Conf. on Computer Communications (INFOCOM)*, 2015, pp. 316–324.

[10] P. Narang, C. Hota, and H. T. Sencar, “Noise-Resistant Mechanisms for the Detection of Stealthy Peer-to-Peer Botnets,” *Computer Communications*, 2016, in press.

[11] R. R. R. Barbosa, R. Sadre, and A. Pras, *Difficulties in Modeling SCADA Traffic: A Comparative Analysis*, 2012, pp. 126–135.

[12] D. Germanus, A. Khelil, and N. Suri, “Increasing the Resilience of Critical SCADA Systems Using Peer-to-Peer Overlays,” in *Proc. of the 1st Int. Conf. on Architecting Critical Systems*, 2010, pp. 161–178.

[13] D. Germanus, A. Khelil, J. Schwandke, and N. Suri, “Coral: Reliable and Low-Latency P2P Convergecast for Critical Sensor Data Collection,” in *Proc. of the 2013 IEEE Int. Conf. on Smart Grid Communications (SmartGridComm)*, 2013, pp. 300–305.

[14] P. Narang, S. Ray, C. Hota, and V. Venkatakrishnan, “PeerShark: Detecting Peer-to-Peer Botnets by Tracking Conversations,” in *Proc. of the 2014 IEEE Security and Privacy Workshops*, 2014, pp. 108–115.

[15] C. Rossow, D. Andriesse, T. Werner, B. Stone-Gross, D. Plohmann, C. J. Dietrich, and H. Bos, “SoK: P2PWNED - Modeling and Evaluating the Resilience of Peer-to-Peer Botnets,” in *Proc. of the 2013 IEEE Symp. on Security and Privacy (SP)*, 2013, pp. 97–111.

[16] T. Wüchner, M. Ochoa, M. Golagha, G. Srivastava, T. Schreck, and A. Pretschner, “MalFlow: Identification of C&C Servers Through Host-Based Data Flow Profiling,” in *Proc. of the 31st Annu. ACM Symp. on Applied Computing*, 2016, pp. 2087–2094.

[17] G. Kirubavathi and R. Anitha, “Botnet Detection via Mining of Traffic Flow Characteristics,” *Computers & Electrical Engineering*, vol. 50, pp. 91–101, 2016.