Hiding Critical Targets in Smart Grid Networks

Wei Bao, Qinghua Li
Department of Computer Science and Computer Engineering,
University of Arkansas
Email: {weibao, qinghual}@uark.edu

***Abstract*—With the integration of advanced communication technologies, the power grid is expected to greatly enhance efficiency and reliability of future power systems. However, since most electrical devices in power grid substations are connected via communication networks, cyber security of these communication networks becomes a critical issue. Real-World incidents such as Stuxnet have shown the feasibility of compromising a device in the power grid network to further launch more sophisticated attacks. To deal with security attacks of this spirit, this paper aims to hide critical targets from compromised internal nodes and hence protect them from further attacks launched by those compromised nodes. In particular, we consider substation networks and propose to add carefully-controlled dummy traffic to a substation network to make critical target nodes indistinguishable from other nodes in network traffic patterns. This paper describes the design and evaluation of such a scheme. Evaluations show that the scheme can effectively protect critical nodes with acceptable communication cost.**

***Index Terms*—Power Grid, Cyber Security, Dummy Traffic**

I. Introduction

Power grid communication networks suffer from network intrusions [1]. Real-World incidents such as Stuxnet have shown that an adversary can compromised an internal device, and do what the compromised internal device can do. In particular, one attack through a compromised internal node is to eavesdrop communications on the network and identify critical targets for advanced attacks (e.g., those for stealthy false data injection attacks) through traffic analysis. Despite many peripheral security technologies such as firewall and intrusion detection systems have already been deployed in many utilities, passive espionage from an internal node cannot be addressed by peripheral technologies. The compromised node can gradually learn the communication patterns of nodes in the network and map them to critical target nodes whose communication patterns are known in advance.

In this paper, we consider a substation network and propose to add dummy traffic between nodes so as to make nodes traffic patterns indistinguishable from each other. This problem is nontrivial since adding dummy traffic increases the communication cost and thus we want to achieve the defense performance without incurring too much communication cost. In one naive solution, one can add much dummy traffic so that every node communicates with every other node in the same pattern, so that all nodes appear the same. However, this solution has very high communication cost. To reduce the cost, we borrow ideas from the privacy research community, and try to achieve k-anonymity for each node. Specifically, we divide nodes into groups where each group has at least k nodes and those nodes have similar communication patterns. In this way, we achieve a tradeoff between protection of critical targets and reduction of communication cost. Although k-anonymity is a concept in privacy research, it is used here to address cybersecurity problems.

The contribution of this paper is summarized as follows:

- We design a novel k-anonymity based scheme to make the communication patterns of critical nodes indistinguishable from other nodes and hence hide critical targets from espionage by even compromised nodes in the same network, preventing further sophisticated cyber attacks. To the best of our knowledge, this solution is the first of its kind for power grid networks.
- We implement the proposed scheme in a simulator and perform extensive simulations to evaluate the scheme's performance and cost.

The rest of the paper is organized as follows. Section II reviews related work. Section III describes the design of k-anonymity system. Section IV shows evaluation results. Section V concludes the paper.

II. Related Work

In the smart grid domain, the most related work to this paper is Moving target defense (MTD) solutions that also increase the difficulty of attacks by increasing system dynamics.

Groat et al. [2] propose a scheme to secure power grid communication networks using IPv6 Moving Target Defense technique. The key principle behind their work is to dynamically change the IP addresses of devices. However, changing IP address only is not enough since the attacker can still identify critical targets from the communication characteristics in other layers, e.g., MAC addresses and application-layer characteristics.

Peng et al. [3] propose a method to change cloud infrastructure configuration constantly so as to confuse attackers without significantly degrading the quality of service. However, this method is designed for cloud infrastructure, not for power grid system. Carter et al. [4] propose an approach to protect operating system security by constantly changing the properties of operating systems. Yuan et al. [5] prevent security attacks of software systems by using adaptive security methods.

Al [6] proposes a new moving target defense architecture to hinder the adversary's capabilities in scanning or discovering network targets, launching DoS attacks and creating botnets structure. The system dynamically changes IP address and packet routes. Kil et al. [7] design an address space layout permutation system. The system constantly permutes critical memory regions to secure data segment. Williams et al. [8] develop a genesis software development tool chain to secure the application level of virtual machine. However, none of the above approaches can be directly applied to protect power grid communication networks.

III. System Design

In a substation network, there exists normal communication traffic among different network nodes, such as SCADA traffic and communications between protective relay and circuit breaker. These network traffic could be exploited to infer critical target nodes (e.g., circuit breaker) in the network through traffic analysis. Unfortunately, the Ethernet currently widely used in many substations does not prevent a node from learning about other nodes through eavesdropping and analyzing network traffic. To address this problem, we propose a scheme that can achieve *source k-anonymity* for hiding a node's communication patterns; i.e. for every node in the network, there always exists at least k-1 other nodes which share similar communication patterns. In this way, identifying the critical target becomes a much more difficult task for the attacker.

A. System Structure

As shown in Figure 1, there are two types of nodes, *virtual nodes* which do not really exist and *real nodes* which consists of all the real nodes in the network. Dotted lines represent dummy traffic between virtual nodes and real nodes, and solid lines represent real traffic among real nodes. The idea is to divide nodes into groups where there are at least k nodes inside each group and the nodes inside the same group share the same traffic patterns. Dummy traffic is added as needed to implement k-anonymity of traffic.

First of all, for each node N in the real group with S outgoing connections and R incoming connections, if $(i-1)*n \leq S \leq i*n$, where $i = 1, 2, \cdots, m-1, m$, m and n are both integer system parameters, randomly choose $[(i+j)*n - S]$ nodes in the virtual group with probability of $(m-i+1-j)^3 / \sum_{x=1}^{m-i+1} x^3$ and send dummy messages to those nodes so as to change S to $(i+j)*n$, where $j = 0, 1, \cdots, m-i$. The same rule is applied to R. For simplicity, the probabilistic function we adopt and found effective is $f(j) = (m-j)^3 / \sum_{a=1}^{m} a^3$, where $j = 0, 1, \cdots, m-1$.

Next, check total number of nodes in each range of $(i-1)*n$ and $i*n$, where $i = 1, 2, \cdots, m-1, m$. If the total number of nodes is less than k in range of $(i-1)*n$ and $i*n$, merge the nodes in this range with nodes in range of $i*n$ and $(i+1)*n$ and keep merging up until the total number of nodes after merging is greater than or equal to k, namely, generating extra dummy traffic connections to ensure that there are at least k nodes in each range.

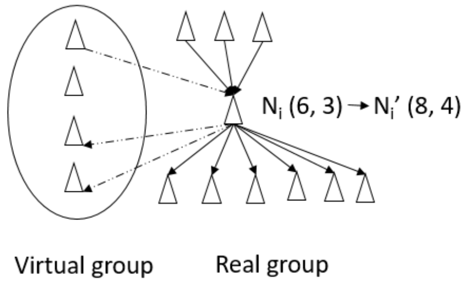

Fig. 1: System Architecture

B. Inter-Node Communication Pattern

In the power substation networks, there exists some easily recognizable communication patterns between different nodes, such as relays communicating with circuit breakers. An inside attacker can observe these communication traffic and infer critical target nodes via traffic analysis. To hide those communication patterns between different pairwise connections from an inside attacker, generating dummy messages between pairwise connections could be an effective solution.

Let the inter-message delay (imd) between message $i (i > 0)$ and $i+1$ from some pair node connection be $imd_i = t_{i+1} - t_i$, where t_i is the transmission time of message i between the pairwise connection. An inside attacker can observe a sequence of continuous inter-message delays, which can be represented by a distribution $X = \{imd_1, imd_2, \cdots\}$. Ideally, inter-message delays from all pairwise connections follow the same distribution. In our case with statistically strong guarantee, distributions of inter-message delays are actually statistically indistinguishable from each other.

The property of statistically indistinguishable distribution is related to two security parameters α and ϵ, where α measures the goodness of fit to a targeted probabilistic distribution and ϵ measures how close of the parameter derived from the observations to that of the population [9]. These two security parameters are used together so that message intervals from different pairwise connections follow similar distributions and an inside attacker cannot tell the difference through a statistical hypothesis test.

Specifically, Algorithm 1 implements our dummy traffic generation method. To maximize the communication randomness and to simplify the problem, we choose the modified exponential distribution to control the rate of dummy traffic generation; i.e., the inter-message delays in different connections follow the same exponential distribution [10]. Anderson-Darling test (A-D test in short) [11] is used to guarantee statistical indistinguishability of the inter-message delay distribution. A-D test is a goodness of fit test to determine if a series of data follow a certain probabilistic distribution. Because A-D test is a statistical test, the solution to pass the test is not unique. Therefore, A-D test is repeated until the test is passed.

Algorithm 1 Search Largest Inter-Message Delay for Dummy Messages

```
Input: mean $\mu$ and a sequence of inter-message time intervals
    $(imd_i (0 < i < n))$
Output: a time interval $imd$ following the modified exponential distribution with mean $\mu$;
1: seed($seed$); {Assign the seed for random number generation to each node}
2: STEPSIZE = rand(0, first quartile of time intervals);
3: repeat
4:     $imd$ -= STEPSIZE; {Anderson-Darling test begins from some positive large number}
5:     $ret$ = A-D($\{imd_1, imd_2, \cdots, imd_{n-1}, imd\}$);
6: until $ret$==TRUE
7: return $imd$;
```

Since the generation of dummy traffic will add extra network overhead, we pursue the largest possible inter-message delay that still passes A-D test for dummy traffic generation.

For a real message that needs to be sent, to avoid violating the exponential distribution, the message might need to wait a short time before being sent out. We want to minimize the waiting time so as to minimize our scheme's impact on real messages. Algorithm 2 shows how we obtain the smallest possible inter-message delay (i.e., waiting time) for real message traffic that can still pass A-D test.

In addition, to make dummy messages more like real messages, similar to message interval, messages will be padded to make message size follow the same exponential distributions to achieve indistinguishability. The details are omitted here for conciseness.

Algorithm 2 Search Smallest Inter-Message Delay for Real Messages

```
Input: mean $\mu$ and a sequence of inter-message time intervals
    $(imd_i (0 < i < n))$
Output: a time interval $imd$ following the modified exponential distribution with mean $\mu$;
1: seed($seed$); {Assign the seed for random number generation to each node}
2: STEPSIZE = rand(0, first quartile of time intervals);
3: repeat
4:     $imd$ += STEPSIZE; {Anderson-Darling test begins from 0}
5:     $ret$ = A-D($\{imd_1, imd_2, \cdots, imd_{n-1}, imd\}$);
6: until $ret$==TRUE
7: return $imd$;
```

C. Port Numbers

When there are multiple communication protocols in the network, each communication protocol is associated with a certain unique port number. To achieve k-anonymity regarding port number, suppose the incoming and outgoing connection patterns for each real node are anonymized using the method described above; i.e. there are k nodes with the same message interval and message size distributions. Then, we use port numbers in each node's dummy traffic so that these k nodes share the same set of port numbers in their messages. For example, as shown in Figure 2, node N has a connection on port b, but other nodes in its group do not use port b. Then to achieve k-anonymity in port number, other nodes will also use port b in their dummy traffic.

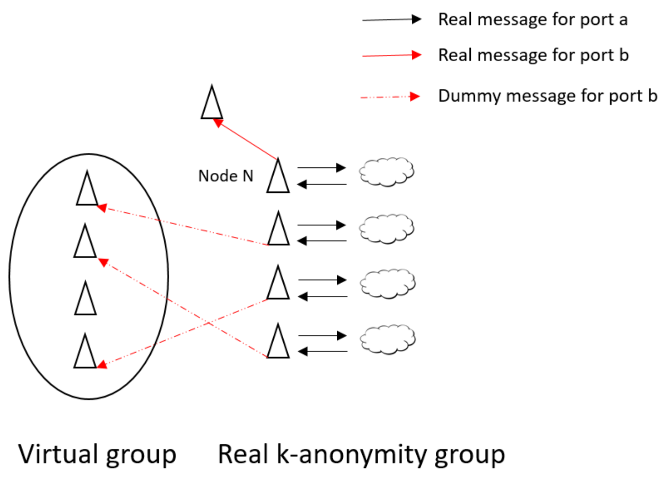

Fig. 2: Multiple Protocol System Structure

IV. Evaluations

A. Experimental Methodology

We implemented our scheme in ns-3 [12] for a simulated substation network and evaluated it with extensive simulations. In our simulation, we generate three types of messages: SCADA messages, real messages (which include other messages than SCADA, e.g., those triggered by abnormal events), and dummy messages. SCADA messages are generated for each node in the simulated network every 4s (which is a typical interval for SCADA data collection). For other real messages, the inter-message delay follows exponential distribution. Dummy messages are generated according to our algorithms.

B. Extra Delays Induced to Real Messages

Since our scheme makes the message intervals of each pairwise connection follow an exponential distribution, a real message might be intentionally delayed a bit to follow the distribution. This set of simulations aim to evaluate the extra delays induced to real messages. Since there is no existing work on this topic, we compare our scheme with a baseline scheme, called *constant rate scheme*, which generates dummy traffic in such a way that the combined traffic (including real messages and dummy messages) are at constant rate. To evaluate their performances, we test different average message intervals of the combined traffic. Different α and ϵ are also tested to control the degree of statistical indistinguishability.

TABLE I: Extra Message Delay When the Average Message Interval for Combined Traffic Is 10s

Real Msg Frequency	20	40	60	80	100	200
Constant Rate (ms)	5270	5680	5960	6110	6290	6310
Our Scheme (ms) ($\alpha = 0.05, \epsilon = 0.01$)	600	450	390	310	250	190
Our Scheme (ms) ($\alpha = 0.1, \epsilon = 0.2$)	420	380	270	200	110	80

TABLE II: Extra Message Delay When the Average Message Interval for Combined Traffic Is 1s

Real Msg Frequency	20	40	60	80	100	200
Constant Rate (ms)	508	468	486	601	529	509
Our Scheme (ms) ($\alpha = 0.05, \epsilon = 0.01$)	230	190	150	110	60	50
Our Scheme (ms) ($\alpha = 0.1, \epsilon = 0.2$)	1.1	0.9	0.8	0.8	0.6	0.5

We run the simulation for 1800s and get the average delay for real messages under different parameter settings. Table I and Table II show the results of extra message delays when the average message interval of the combine traffic is 10s and 1s respectively. It is observed that the extra message delay caused by our scheme is much lower than that caused by the baseline scheme. We also found that for our scheme, the extra message delay decreases as α or ϵ increases (i.e., the requirement for indistinguishability is relaxed) which is reasonable.

C. Total Number of Messages

This part evaluates the communication cost in the total number of messages transmitted. Table III shows the total number of messages when the system parameter n is 4. Compared with the baseline scheme, our scheme saves many messages and thus has much lower communication cost. Also, compared with the original traffic without any protection, our scheme only increases the cost by at most 50% in those cases, which is not a big problem for substations with not-quite-frequent SCADA traffic and powerful Ethernet switches. Also, when ϵ is fixed but α increases (or is relaxed), the number of messages in our scheme decreases (see row 1-3, and row 7-9). Similarly, when α is fixed but ϵ increases (or is relaxed), the number of messages in our scheme decreases as well (see row 4-6, and row 10-12).

We also did simulations when the parameter n is 5 and 6 and the results are shown in Table IV and Table V respectively. The trend is similar with the case where n is 4. It can also be seen that as system parameter n increases, the total communication cost increases. This is due to larger groups caused by larger n which induces more extra dummy traffic.

TABLE III: Total Number of Messages When n = 4

$n = 4$	Original	Constant Rate	Our Scheme
Combined Avg Interval = 10s, $\alpha = 0.01, \epsilon = 0.1$	19871	24589	21132
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.1$	19871	24589	20976
Combined Avg Interval = 10s, $\alpha = 0.1, \epsilon = 0.1$	19871	24589	20124
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.01$	19871	24589	22320
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.1$	19871	24589	21410
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.2$	19871	24589	20342
Combined Avg Interval = 1s, $\alpha = 0.01, \epsilon = 0.1$	19871	58793	30132
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	19871	58793	29854
Combined Avg Interval = 1s, $\alpha = 0.1, \epsilon = 0.1$	19871	58793	28850
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.01$	19871	58793	30987
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	19871	58793	29120
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.2$	19871	58793	27871

TABLE IV: Total Number of Messages When n = 5

$n = 5$	Original	Constant Rate	Our Scheme
Combined Avg Interval = 10s, $\alpha = 0.01, \epsilon = 0.1$	19871	26543	23154
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.1$	19871	26543	22412
Combined Avg Interval = 10s, $\alpha = 0.1, \epsilon = 0.1$	19871	26543	22031
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.01$	19871	26543	23879
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.1$	19871	26543	22990
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.2$	19871	26543	22311
Combined Avg Interval = 1s, $\alpha = 0.01, \epsilon = 0.1$	19871	60121	31850
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	19871	60121	31123
Combined Avg Interval = 1s, $\alpha = 0.1, \epsilon = 0.1$	19871	60121	30998
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.01$	19871	60121	32412
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	19871	60121	31952
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.2$	19871	60121	31540

D. Extra Message Size

Since our scheme makes the message size follow exponential distributions, some messages must be padded to meet the requirement which means larger message size. This part evaluates the extra message size induced by our scheme. We generate real messages whose sizes follow a Uniform distribution in the range from 20 to 80 bytes (Case I), and a Uniform distribution in the range from 700 to 1500 bytes (Case II). Table VI and VII show the average message size for case I and II respectively. We can observe that the average message size decreases as the indistinguishability parameters α and ϵ are relaxed. The extra average message size induced

TABLE V: Total Number of Messages When n = 6

$n = 6$	Original	Constant Rate	Our Scheme
Combined Avg Interval = 10s, $\alpha = 0.01, \epsilon = 0.1$	19871	29621	25476
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.1$	19871	29621	24875
Combined Avg Interval = 10s, $\alpha = 0.1, \epsilon = 0.1$	19871	29621	24110
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.01$	19871	29621	26133
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.1$	19871	29621	25140
Combined Avg Interval = 10s, $\alpha = 0.05, \epsilon = 0.2$	19871	29621	23871
Combined Avg Interval = 1s, $\alpha = 0.01, \epsilon = 0.1$	19871	62547	33110
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	19871	62547	32661
Combined Avg Interval = 1s, $\alpha = 0.1, \epsilon = 0.1$	19871	62547	32102
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.01$	19871	62547	33897
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	19871	62547	32990
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.2$	19871	62547	31982

TABLE VI: Average Message Size for Case I

	Original (bytes)	Our Scheme (bytes)
Combined Avg Interval = 1s, $\alpha = 0.01, \epsilon = 0.1$	53	61
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	53	59
Combined Avg Interval = 1s, $\alpha = 0.1, \epsilon = 0.1$	53	58
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.01$	53	62
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	53	61
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.2$	53	59

TABLE VII: Average Message Size for Case II

	Original (bytes)	Our Scheme (bytes)
Combined Avg Interval = 1s, $\alpha = 0.01, \epsilon = 0.1$	1120	1295
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	1120	1288
Combined Avg Interval = 1s, $\alpha = 0.1, \epsilon = 0.1$	1120	1283
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.01$	1120	1302
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.1$	1120	1294
Combined Avg Interval = 1s, $\alpha = 0.05, \epsilon = 0.2$	1120	1289

by our scheme is less than 18% compared to original system.

V. Conclusion

This paper proposed a dummy traffic-based solution to hide critical target nodes from network espionage by an inner compromised node. We evaluated the scheme's performance and cost through simulations. Experimental results show that the system can effectively hide critical nodes with insignificant impact to the real system.

Acknowledgment

This work was supported in part by the Department of Energy under Award Number DE-OE0000779.

References

[1] W. Wang and Z. Lu, "Cyber security in the smart grid: Survey and challenges," *Computer Networks*, vol. 57, no. 5, pp. 1344–1371, 2013.

[2] S. Groat, M. Dunlop, W. Urbanksi, R. Marchany, and J. Tront, "Using an ipv6 moving target defense to protect the smart grid," in *Innovative Smart Grid Technologies (ISGT), 2012 IEEE PES*. IEEE, 2012, pp. 1–7.

[3] W. Peng, F. Li, and X. Zou, "Moving target defense for cloud infrastructures: Lessons from botnets," in *High Performance Cloud Auditing and Applications*. Springer, 2014, pp. 35–64.

[4] K. M. Carter, H. Okhravi, and J. Riordan, "Quantitative analysis of active cyber defenses based on temporal platform diversity," *arXiv preprint arXiv:1401.8255*, 2014.

[5] E. Yuan, N. Esfahani, and S. Malek, "A systematic survey of self-protecting software systems," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 8, no. 4, p. 17, 2014.

[6] E. Al-Shaer, "Toward network configuration randomization for moving target defense," in *Moving Target Defense*. Springer, 2011, pp. 153–159.

[7] C. Kil, J. Jun, C. Bookholt, J. Xu, and P. Ning, "Address space layout permutation (aslp): Towards fine-grained randomization of commodity software," in *Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual*. IEEE, 2006, pp. 339–348.

[8] D. Williams, W. Hu, J. W. Davidson, J. D. Hiser, J. C. Knight, and A. Nguyen-Tuong, "Security through diversity: Leveraging virtual machine technology," *IEEE Security & Privacy*, vol. 7, no. 1, 2009.

[9] T. W. Anderson and D. A. Darling, "Asymptotic theory of certain" goodness of fit" criteria based on stochastic processes," *The annals of mathematical statistics*, pp. 193–212, 1952.

[10] M. Shao, Y. Yang, S. Zhu, and G. Cao, "Towards statistically strong source anonymity for sensor networks," in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*. IEEE, 2008, pp. 51–55.

[11] M. A. Stephens, "Edf statistics for goodness of fit and some comparisons," *Journal of the American statistical Association*, vol. 69, no. 347, pp. 730–737, 1974.

[12] ns-3. [Online]. Available: https://www.nsnam.org/

[13] J. L. Romeu, "Kolmogorov-simirnov: A goodness of fit test for small samples," *START: Selected Topics in Assurance Related Technologies*, vol. 10, no. 6, 2003.