A Time Series Approach for Inferring Orchestrated Probing Campaigns by Analyzing Darknet Traffic

Elias Bou-Harb, Mourad Debbabi, Chadi Assi
NCFTA Canada & Concordia University
Montreal, Quebec, Canada
(e_bouh, debbabi, assi)@encs.concordia.ca

***Abstract*—This paper aims at inferring probing campaigns by investigating darknet traffic. The latter probing events refer to a new phenomenon of reconnaissance activities that are distinguished by their orchestration patterns. The objective is to provide a systematic methodology to infer, in a prompt manner, whether or not the perceived probing packets belong to an orchestrated campaign. Additionally, the methodology could be easily leveraged to generate network traffic signatures to facilitate capturing incoming packets as belonging to the same inferred campaign. Indeed, this would be utilized for early cyber attack warning and notification as well as for simplified analysis and tracking of such events. To realize such goals, the proposed approach models such challenging task as a problem of interpolating and predicting time series with missing values. By initially employing trigonometric interpolation and subsequently executing state space modeling in conjunction with a time-varying window algorithm, the proposed approach is able to pinpoint orchestrated probing campaigns by only monitoring few orchestrated flows. We empirically evaluate the effectiveness of the proposed model using 330 GB of real darknet data. By comparing the outcome with a previously validated work, the results indeed demonstrate the promptness and accuracy of the proposed approach.**

I. Introduction

Probing activities continue to be a growing cyber security concern due to the fact that they are the primary stage of an intrusion attempt that enables an attacker to remotely locate, target, and subsequently exploit vulnerable systems [1]. Such activities could be defined by the task of scanning enterprise networks or Internet wide services, searching for vulnerabilities or ways to infiltrate IT assets. Basically, they render a core technique and a main facilitator of the above mentioned and other cyber attacks. For instance, hackers have employed probing techniques to identify numerous misconfigured proxy servers at the New York Times to access a sensitive database that disclosed more than 3,000 social security numbers. More alarming, a recent incident reported that attackers had escalated a series of "surveillance missions" against cyber-physical infrastructure operating various energy firms that permitted the hackers to infiltrate the control-system software and subsequently manipulate oil and gas pipelines. Hence, it is not surprising that Panjwani et al. [2] concluded that a staggering 50% of attacks against cyber systems are preceded by some form of probing activity. Therefore, the capability to infer and attribute probing activities is a very important task to achieve, as this will aid in preventing cyber attacks from occurring or vulnerabilities from being exploited. Indeed, this paper tackles a very recent phenomenon of probing events dubbed as probing campaigns. Such orchestrated malicious probing events are postulated to be ominously leveraged to cause drastic Internet-wide and enterprise impacts as precursors of various amplified, debilitating and disrupting cyber attacks including, but not limited to, distributed and reflective denial of service attacks, advanced persistent threats and spamming campaigns. Thus, motivated by the imperative requirement to infer such probing campaigns, we frame the paper's contribution as follows:

- Proposing an approach that models the complex problem of inferring probing campaigns as the task of interpolating and predicting time series in the presence of missing values. The model allows the pinpointing of orchestrated campaigns by observing just few probing packets, rendering it very prompt. Further, by only keeping a record of the probing time series, the model is efficient and lightweight in terms of memory and processing requirements, which makes it applicable to be implemented in real-time on operational data streams. To the best of our knowledge, this is the first systematic model that is specifically tailored towards the goal of inferring probing campaigns by observing the dark Internet Protocol (IP) space.
- Uniquely employing time series interpolation and prediction approaches based on trigonometric and state space modeling techniques, namely, the discrete Fourier transform and kalman filter, to tackle this problem.
- Evaluating the promptness and accuracy of the proposed model using 330 GB of real darknet data in addition to validating the model's outcome and advantages by comparing it with a previous work.

The remainder of this paper is organized as follows. In the next section, we provide relevant background information related to darknets and probing campaigns. The problem formulation coupled with the proposed model are presented in Section III. Section IV empirically evaluates the proposed approach, compares its advantages with previous work and validates its inferences. In Section V, we review the related work on various concerned topics. Finally, Section VI summarizes the paper and paves the way for future work.

II. Background

In this section, we provide relevant background information related to darknets and probing campaigns.

978-1-4673-6590-1/15 $31.00 © 2015 IEEE
DOI 10.1109/ARES.2015.9

A. Darknets

A darknet, also commonly referred to as a network telescope or an Internet sink, is a set of routable and allocated yet unused IP addresses [3]. It represents a partial view of the entire Internet address space. From a design perceptive, darknets are transparent and indistinguishable compared with the rest of the Internet space. From a deployment perspective, it is rendered by network sensors that are implemented and dispersed on numerous strategic points throughout the Internet. Such sensors are often distributed and are typically hosted by various global entities, including Internet Service Provides (ISPs), academic and research facilities and backbone networks. The aim of a darknet is to provide a lens on Internet-wide malicious traffic; since darknet IP addresses are unused, any traffic targeting them represents a view of anomalous unsolicited traffic. A network telescope is indeed an effective approach to infer various Internet-scale probing activities [4].

B. Probing Campaigns

Lately, there has been a noteworthy shift towards a new phenomenon of probing events which, throughout this paper, is referred to as probing campaigns. These are distinguished from previous probing incidents as (1) the population of the participating bots is several orders of magnitude larger, (2) the target scope is generally the entire IP address space, and (3) the bots adopt well-orchestrated, often botmaster-coordinated, stealth scan strategies that maximize targets' coverage while minimizing redundancy and overlap. Very recently, Dainotti et al. from the Cooperative Association for Internet Data Analysis (CAIDA) presented a pioneering measurement and analysis study of a 12-day Internet-wide probing campaign targeting VoIP (SIP) servers. In another work [5], the same authors admitted that they have detected the reported SIP campaign, including the malware responsible for its actions (i.e., Sality malware), "serendipitously" (i.e., luckily and accidentally) while analyzing a totally unrelated phenomenon. They also stated that since *currently there exist no cyber security capability to discover such large-scale probing campaigns*, other similar events targeting diverse Internet and organizational infrastructure are going undetected. In another inquisitive, well executed work, an "anonymous" presented and published online [6] what they dubbed as the "Carna Botnet". The author exploited poorly protected Internet devices, developed and distributed a custom binary, to generate one of the largest and most comprehensive IPv4 census ever. The aforementioned two campaign studies differ on various key observations. The work by Dainotti et al. disclosed that the bots were recruited into the probing botnet by means of a new-generation malware while the Carna Botnet was augmented using a custom code binary. Moreover, Dainotti et al. discovered that the bots were coordinated by a botmaster in a command-and-control infrastructure where the bots used a reverse IP-sequential strategy to perform their probing, while the Carna Botnet was C&C-less and its bots used an interleaving permutation method to scan its targets. Further, the work by Dainotti et al. documented a horizontal scan that targeted world-wide SIP servers, while the Carna Botnet did not focus on one specific service but rather attempted to retrieve any available information that was associated with any host and/or service. In this work, we aim at devising a prompt and a lightweight systematic methodology that exploits probing time series to infer orchestrated probing campaigns by observing the darknet IP space. It might be beneficial to mention at this point of the paper that, for the past three years, we have been receiving, on a daily basis, raw darknet data from a trusted third party, namely, Farsight Security [7]. Such traffic originates from the Internet and is destined to numerous /13 network sensors. The data mostly consists of unsolicited TCP, UDP and ICMP traffic. In this work, we will be leveraging a portion of such data to validate our proposed model.

III. Proposed Approach

In this work, we uniquely model the task of inferring orchestrated probing campaigns as the problem of interpolating and predicting time series in the presence of missing values. The core rationale behind this approach stems from the idea that if the probing flow time series demonstrates positive predictability features, thus identifying certain probing patterns, then it might be part of an orchestrated event. To realize such rationale, the proposed approach (1) fingerprints independent probing flows as perceived by the telescope, (2) clusters and builds the probing flow time series, (3) interpolates the time series for data completion purposes and (4) predicts the time series in order to infer orchestrated flows and eliminate independent (i.e., non-orchestrated) ones. In the sequel, we elaborate on the latter four steps.

A. Fingerprinting Independent Flows

In [8], we proposed a new approach, specifically tailored to operate on telescope data, in order to fingerprint probing activities. The approach aimed at detecting the probing activity and identifying the exact technique that was employed in the activity. When empirically evaluated using a significant amount of real telescope data, the approach yielded 0 false negative in compassion with two leading network intrusion detection systems, namely, Snort and Bro. Readers that are interested in more details related to the approach are kindly referred to [8]. In this work, and to successfully extract independent probing flows as perceived by the network telescope, we adopt and leverage the previously proposed approach.

B. Flow Clustering & Time Series Generation

Probing flow packets targeting the darknet have the following 8-tuple form

$$\langle t, src_{ip}, dst_{ip}, src_p, dst_p, prot, ttl, flags \rangle$$

representing the timestamp, the source and destination IP addresses and ports, the transport protocol used, the time-to-live (ttl) value and the packet flags. From recent probing campaign incidents, it was observed that orchestrated probes possess similar values for dst_p, $prot$ and $flags$. Thus, we first amalgamate all the probing flows into different clusters sharing similar values for those packet features. Additionally, based on those reported events, it was demonstrated that a particular campaign has been generated by the same type of operating system; in one work, it was inferred that 97% of the orchestrated probing flows were originating from windows machines while in another, it was revealed that all the flows were generated from Unix environments. Thus, it is desirable that we further cluster the previous groups by the same originating operating system. To achieve this, we investigate

the ttl value of the probing packets as perceived by the network telescope. According to [9], most modern operating systems use only a few selected initial ttl values, particularly, 30, 64, 128, and 255. It is evident that most of these initial ttl values are far apart. Moreover, since Internet traces have shown that few Internet hosts are apart by more than 30 hops, which is also confirmed by our own recent observations, one can determine the initial ttl value of a probing packet that is received at the telescope by selecting the smallest initial value in the set that is larger than its final ttl. For example, if the observed final ttl value is 112, the initial ttl value would be 128. Thus, based on this heuristical approach, we further refine the clusters by subdividing the probing flows according to similar ttl values. Recall that the latter aims at further clustering the groups based on similar originating operating system. It is important to note that probing sources/attackers can use, in theory, certain tools that can explicitly modify the ttl value in an attempt to evade being correctly clustered. For example, `traceroute` is a common application which can set the IP ttl to any random number, independent of the operating systems' kernel's default. However, a probing source who overrides IP ttl to avoid detection should enforce a stable mapping of IP source to IP ttl, in order to prevent us from seeing a noisy IP ttl originating from its IP address, and thus subsequently flagging his IP source as ttl spoofing. In practice, in our experiments, we did not notice any ttl spoofing, where all the inferred ttl values were from those 4 default categories. Please recall, in a nutshell, that the output of the above are independent probing flows clustered based on similar dst_p, $prot$, $flags$ and ttl values. For each of the above independent probing flows within those clusters, we generate their corresponding time series of the form $< t, dst\_ip >$.

Note that due to the simplicity of such time series, the approach is able to generate around 200 thousand of those in under 5 seconds. Further, since the time series is composed of just two packet features, namely, the timestamp and the destination darknet IP address, the storage requirements are minimal; less than 300 MB to save those 200 thousand time series. It is also noteworthy to mention that such storage requirement will not augment by time, since the approach discards the previously extracted time series after processing.

C. Time Series Interpolation

The above time series could be referred to as a time series with missing values in which some of its rows are not captured. This is because (1) the network telescope only covers a portion of the Internet space and thus is not able to perceive all the probes at all times and (2) some probing packets arriving at the telescope could be distorted and thus are filtered out before the generation of the time series. Indeed, without a complete view of the probing time series, it is very difficult, if not impossible, to devise approaches to infer orchestration. To deal with this issue, we resort to time series interpolation. This refers to numerical analysis techniques that aim at constructing new data points from and within a range of a discrete set of known data points. Although there exists several interpolation techniques in the literature, in this work, we employ and tailor, for accuracy and efficiency reasons, trigonometric interpolation, namely, the discrete Fourier transform (dFt) [10]. We omit the theory of the dFt that could be found in [10]. Given a set of $(n+1)$ data points in the probing time series, dFt provides an interpolating function that is composed of finite sum of cosines and sine that connects all $(n + 1)$ data points, including any missing values in between. However, for the previous function to be directly applicable to the extracted probing time series, we modify the dst_{ip} packet feature in the time series from the typical IPv4 CIDR notation to a discrete numeric by using a simple developed mechanism.

Algorithm 1: A time-varying window algorithm to distinguish between orchestrated and independent probing flows by leveraging kalman's error covariance

```
Data: Probing Clusters, PC,
Probing Time Series, PTS
Result: List of Orchestrated Flows, OF
for PTS in PC do
    Processing_Window, PW=5;
    Comparison_Window, CW=1;
    while PTS in PW do
        while PTS in CW do
            v1=kalman.execute(PTS);
            CW++;
            v2=kalman.execute(PTS);
            if v1 > v2 then
                flag((CW- -).FirstElement();
                OF.add(CW- -).OtherElements();
            end
            if v1 < v2 then
                flag(CW).LastElement());
                OF.add(CW).OtherElements();
            end
            CW++;
        end
    end
    PW+=5;
end
```

D. Time Series Prediction

After interpolating the probing time series within the clusters, the next step would be to verify if such series indicate any signs of predictability. If confirmed, this would provide evidence that the probing flows indeed possess some orchestration logic. To achieve that task, we tailor and employ a recursive optimal stochastic estimator, namely, the kalman filter [11]. Although we have selected to leverage this specific theoretical-based technique because of various reasons, including, (1) its superior results in practice due to optimality and structure, (2) its convenient form that permits online real-time processing, (3) its ease to formulate and implement and (4) its efficiency when calculating the measurement equations and the error covariances, the main reason, however, would be due to its significant applicability to the problem in-hand; the kalman filter infers parameters of interest from indirect, inaccurate and uncertain observations. This is coherent with our problem since the time series under predictability verification has been interpolated and thus might contain some uncertain records. The inner workings of the kalman filter could be found in [11]. Going back to our probing time series, recall that the aim is to infer whether or not the probes demonstrate any signs of predictability and thus orchestration. We exploit the kalman filtering error covariance in conjunction with a time-varying

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Probed Destination Port	5060	23	80	1433	443
Transport Protocol	UDP	TCP	TCP	TCP	TCP
Probing Flags	UDP	TCP SYN	TCP SYN	TCP ACK	TCP FIN
TTL	128	128	64	64	128
Number of Probing Flows	2698	3067	913	2983	939

TABLE I: Summary of the clustered probing flows

window algorithm to achieve the latter. The pseudocode of the algorithm is presented next. Algorithm 1 operates on the basis of two time windows. The first is used to load the probing time series within each cluster into volatile memory for processing while the second is used to compare the probing time series. The comparison is based on kalman's error covariance; the algorithm flags those flows that increase the error as demonstrating non-predictability while inferring orchestrated ones by monitoring a decrease in the error metric. From a complexity perceptive, the algorithm requires O(m) space complexity where m is the size of the probing time series within each cluster and O($m + n + p$) time complexity where n is the size of the probing time series related to the comparison window and p is the required time for the kalman filter to process the time series within q fixed iterations. It is noteworthy to mention that in this work, we have chosen a q value of 10 iterations for the kalman filter in order to measure and compare the error covariance. Note that the choice of the value per say is not an issue; rather, its systematic application for all the probing time series for effective and accurate comparison. Further, as mentioned in Section III-B, recall that since the probing time series is very simple, there is no compelling reason (i.e., from a memory or processing requirements) to optimize the values of the window sizes. In practice, the algorithm can process and decide upon a time series with 1 thousand records in less than 10 seconds and require, on average, around 10 MB in volatile storage.

IV. Empirical Evaluation

As mentioned in Section II-B, we possess raw darknet/telescope data that we receive from a trusted third party, namely, Farsight Security [7]. Such traffic originates from the Internet and is destined to numerous /13 network sensors. In this section, we leverage 330 GB of such data extracted from the month of April 2014 to validate the promptness and accuracy of the proposed model. Consistent with Section III-A, we infer 30 thousand unique independent probing flows. Further, coherent with Section III-B, around 35% of the latter were successfully clustered into 5 groups. The details of those clusters are summarized in Table I. It can be noted that the majority of the probing flows have been generated by Windows machines. For each probing flow within each of the 5 clusters, we generate their probing time series in accordance with Section III-B. Figure 1 shows the CDF of the probed destinations within the first three clusters. From the Figures, we can infer that cluster 2 is the most dispersed; 50% of the sources probed more than 1300 destinations. Further, we can extract that clusters 3 is more focused in which most of its sources probed a small number of destinations. We proceed by executing the discrete-time Fourier transform interpolation approach on the time series within each of the 5 clusters as indicated in Section III-C. Figure 2 corroborates the fact that the probing flows of cluster 2 are indeed dispersed; not only they target a significant amount of destinations as was inferred from Figure 1, but also most of them were not captured by our leveraged /13 darkspace, which is indicated by the large percentage of interpolated multiple missing value time series. Subsequently, consistent with Section III-D, we invoke

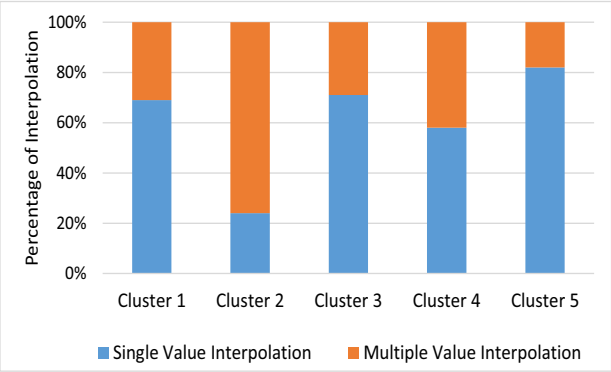
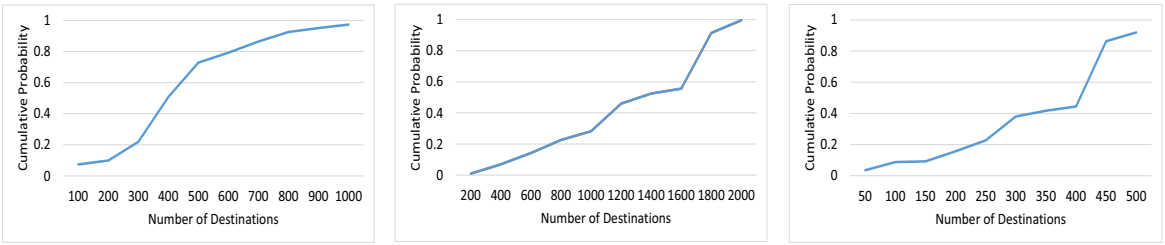


Fig. 2: Distribution of the Types of the Interpolated Values

the kalman filter coupled with Algorithm 1 to distinguish between orchestrated and non-orchestrated flows. By executing Algorithm 1, the first four clusters of Table I demonstrated positive orchestration behavior. Cluster 5 was eliminated as it revealed negative coordinated behavior. By employing simple tcpdump signatures on the data derived from the features (i.e., destination port, protocol, flags, ttl) of the first four clusters, Table I could be re-represented as orchestrated flow clusters as summarized in Table II. By comparing Tables I and II, one can note the drop between clustered independent flows (Table I) and orchestrated flows (Table II) as achieved by Algorithm 1.

A. Comparison with Previous Work

The first attempt to tackle the problem of inferring large-scale probing campaigns by observing network telescopes was rendered in [12]. In that work, the authors proposed a set of big data behavioral analytics to scrutinize probes as received by the darkspace. The analytics were based on statistical, heuristical and fuzzy hashing approaches that aim at generating feature vectors for each of the perceived probing flows. The rationale is to automatically cluster the probing sources possessing similar feature vectors using data mining techniques



(a) CDF of Destinations in Cluster 1 (b) CDF of Destinations in Cluster 2 (c) CDF of Destinations in Cluster 3

Fig. 1: CDF of the probed destinations within the first three clusters

	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Probed Destination Port	5060	23	80	1433
Transport Protocol	UDP	TCP	TCP	TCP
Probing Flags	UDP	TCP SYN	TCP SYN	TCP ACK
TTL	128	128	64	64
Number of Orchestrated Probing Flows	1567	1863	858	587

TABLE II: Summary of the inferred orchestrated probing campaigns

	C1	**C2**	**C3**
Employed probing technique:	UDP	TCP SYN	ACK
Probing traffic:	Random	Random	Pattern
Employed pattern:	Null	Null	[19.17-21.23]
Adopted probing strategy:	Reverse IP-sequential	Uniform Permutation	Forward IP-Sequential
Nature of probing source:	Bot	Bot	Tool
Type of probing:	Dispersed	Dispersed	Targeted
Signs of malware infection:	Yes	Yes	No
Exact malware type/variant:	Virus.Win32.Sality.bh	Trojan.Win32.Jorik	Null
Probing rate (in pps):	12	118	77
Target port:	5060	80	1433
Number of Probing Bots/Sources:	846	817	438

TABLE III: The Automatically inferred patterns capturing three large-scale orchestrated probing campaigns by employing the approach in [12]

in order to infer the probing campaigns. The outcome of that approach was validated using a third party publically available data set, namely, Dshield's repository. Readers that are interested in more details related to that work, are kindly referred to [12]. To compare and contrast the advantages of the presently proposed work, in this section, we compare its outcome with that previous work. To achieve that, we implemented the previously proposed approach in [12] on the same dataset of this work. The outcome is illustrated in Table III. By comparing Tables II and III, we can pinpoint that both approaches were able to infer orchestrated clusters 1, 3 and 4 of Table II. One difference that is related to those campaigns is concerning the number of identified orchestrated flows; the number of inferred orchestrated probing sources using the proposed approach exceeded those inferred using [12]. We semi-automatically investigated the additional flows as identified using the proposed approach, and we confirmed that they are indeed part of the campaign. Hence, we assert that the approach in [12] missed some flows as belonging to the same campaign. Further, that approach completely missed a fourth campaign, which was pinpointed by our proposed approach as cluster 2 of Table II. Additionally, our proposed approach is not only more accurate but is also more prompt than the previous approach; our proposed approach can flag a campaign by observing only few orchestrated flows while the approach in [12] requires the creation of the complete feature pattern (i.e., processing of all the perceived data) to cluster the sources into well-defined campaigns. Moreover, our proposed approach is more efficient as it only records and process lightweight time series (recall Section III-B) while the approach in [12] needs to maintain, in memory, the entire darknet data flows. Last but not least, the presently proposed approach is more systematic and formal compared with the approach in [12] as it deals with time series, trigonometric interpolation and state space modeling as apposed to relying on the output of statistical tests and heuristics. Since the approach in [12] attributes the campaign to a certain malware family using correlation techniques between probing and malware samples [12], we may port this capability to the our proposed approach, which currently lacks that feature.

V. Related Work

In this section, we briefly highlight on some related work in various concerned topics. In the area of extracting probing events, Li et al. [13] extracted such events from darknet traffic using time series analysis. They further executed manual analysis and visualization techniques to extract the rough boundaries of such events. In the context of analyzing probing events, the same authors presented an analysis that drew upon extensive honeynet data to explore the prevalence of different types of scanning activities. Additionally, they designed mathematical and observational schemes to extrapolate the global properties of scanning events including total population and target scope. In the area of probing measurement studies, in addition to [6], Benoit et al. [14] presented the world's first Web census while Heidemann et al. [15] were among the first to survey edge hosts in the visible Internet. Further, Cui and Stolfo [16] presented a quantitative analysis of the insecurity of embedded network devices obtained from a wide-area scan. Last but not least, a number of botnet detection systems have been proposed in the literature. Some of those investigates specific channels, others might require deep packet inspection or training periods, while the majority depends on malware infections and/or attack lifecycles. To the best of our knowledge and as stated in [5], the capability to infer large-scale orchestrated probing events by solely analyzing the dark IP space does not exist, rendering the proposed approach as a novel contribution.

VI. Conclusion

In this paper, we approached the problem of inferring orchestrated probing campaigns from a time series perspective. The motivation behind that was two-folds. First, from an efficiency perspective, we thought it would be desirable to work with the artifact of the data rather than the data itself. Second, scientifically, it is typically more sound to generate inferences and insights using formal methods and approaches rather than depending on statistical inferences or heuristics. Thus, in this paper, we uniquely modeled the challenging task of inferring probing campaigns as a problem of interpolating and predicting time series in the presence of missing values. Initially, the model extracts independent probing flows as perceived by the telescope. Subsequently, the model builds numerous time series clusters sharing similar traffic features. For data completion purposes, the model employs trigonometric interpolation on such probing time series. To infer possible orchestration behavior, the model executes state space modeling in conjunction with a time-varying window algorithm. Empirical evaluations with significant amount of darknet data indeed demonstrated the promptness and accuracy of the proposed model in comparison with a previous work. As for future work, we plan to perform probing campaign intent analysis; the ability to infer what the probing sources will eventually execute after finalizing their probing activities. We aim to achieve the latter by correlating the generated inferences from this work with other data sources, including but not limited to, passive dns and public intrusion and firewall logs.

Acknowledgment

The authors are grateful for Concordia University and the Natural Sciences and Engineering Research Council of Canada (NSERC) for supporting this work. The first author is supported by the Alexander Graham Bell Canada Graduate Scholarship (CGS) from NSERC.

References

[1] Elias Bou-Harb, Mourad Debbabi, and Chadi Assi. Cyber scanning: a comprehensive survey. *Communications Surveys & Tutorials, IEEE*, 16(3):1496–1519, 2013.

[2] S Panjwani et al. An experimental evaluation to determine if port scans are precursors to an attack. In *Proceedings of the International Conference on DSN 2005*, pages 602–611, 2005.

[3] David Moore, Colleen Shannon, Geoffrey M Voelker, and Stefan Savage. *Network telescopes: Technical report*. Department of Computer Science and Engineering, University of California, San Diego, 2004.

[4] Mark Allman, Vern Paxson, and Jeff Terrell. A brief history of scanning. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 77–82. ACM, 2007.

[5] A. Dainotti, A. King, and K. Claffy. Analysis of Internet-wide Probing using Darknets. In *Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, Oct 2012.

[6] Internet Census 2012-Port scanning /0 using insecure embedded devices. http://tinyurl.com/c8af8lt.

[7] Security Information Exchange (SIE), Farsight Security Inc. https://www.farsightsecurity.com.

[8] Elias Bou-Harb, Mourad Debbabi, and Chadi Assi. On fingerprinting probing activities. *Computers & Security*, 43:35–48, 2014.

[9] Jennifer Rexford, Jia Wang, Zhen Xiao, and Yin Zhang. Bgp routing stability of popular destinations. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 197–202. ACM, 2002.

[10] Antoni Zygmund. *Trigonometric series*, volume 1. Cambridge university press, 2002.

[11] Greg Welch and Gary Bishop. An introduction to the kalman filter, 1995.

[12] Elias Bou-Harb, Mourad Debbabi, and Chadi Assi. Behavioral analytics for inferring large-scale orchestrated probing events. In *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, pages 506–511. IEEE, 2014.

[13] Zhichun Li, A. Goyal, Yan Chen, and V. Paxson. Towards situational awareness of large-scale botnet probing events. *IEEE Transactions on Information Forensics and Security,*, 6(1):175–188, 2011.

[14] Darcy Benoit and André Trudel. World's first web census. *International Journal of Web Information Systems*, 3(4):378, 2007.

[15] John Heidemann et al. Census and survey of the visible internet. In *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, IMC '08, pages 169–182, New York, NY, USA, 2008. ACM.

[16] Ang Cui and Salvatore J. Stolfo. A quantitative analysis of the insecurity of embedded network devices: results of a wide-area scan. In *Proceedings of the 26th Annual Computer Security Applications Conference*, ACSAC '10, pages 97–106, New York, NY, USA, 2010. ACM.