BotTrack: Tracking Botnets Using NetFlow and PageRank

Jérôme François, Shaonan Wang, Radu State, and Thomas Engel

Interdisciplinary Centre for Security, Reliability and Trust (SnT)
University of Luxembourg - Campus Kircherg, L-1359 Luxembourg, Luxembourg
firstname.lastname@uni.lu

Abstract. With large scale botnets emerging as one of the major current threats, the automatic detection of botnet traffic is of high importance for service providers and large campus network monitoring. Faced with high speed network connections, detecting botnets must be efficient and accurate. This paper proposes a novel approach for this task, where NetFlow related data is correlated and a host dependency model is leveraged for advanced data mining purposes. We extend the popular linkage analysis algorithm PageRank [27] with an additional clustering process in order to efficiently detect stealthy botnets using peer-to-peer communication infrastructures and not exhibiting large volumes of traffic. The key conceptual component in our approach is to analyze communication behavioral patterns and to infer potential botnet activities.

Keywords: Botnets, PageRank, Network security.

1 Introduction

The ever-continuing struggle between malware operators and security officers is a perpetual and parallel development of new attack tools and mitigation techniques. Botnets are known to be a major threat [5]. The recent trend in malware design leverages peer-to-peer communication infrastructures and state-of-the-art cryptography that significantly raises the stakes in their detection [14]. While bots participating in denial of service attacks and spam campaigns can be easily detected by traffic analysis, it is much harder to detect stealthy bots deployed for information stealing and espionage purposes. We address this specific type of botnet in our paper. The contributions of our paper are threefold:

1. We propose a NetFlow [7] monitoring framework that leverages a simple host dependency model for tracking communication patterns.
2. Linkage analysis and clustering techniques that identify groups of hosts sharing similar behavioral/communication patterns
3. Experiments using real NetFlow data in order to assess the performance of detecting botnet traffic in NetFlow data.

This work is related to our previous one [37] but the construction of the dependency does not need any prior knowledge about potential risks in this paper.

J. Domingo-Pascual et al. (Eds.): NETWORKING 2011, Part I, LNCS 6640, pp. 1–14, 2011.

Thus, it leads to additional clustering techniques for being suitable. Unlike our previous work, this paper tries to detect large-scale shared behavior and particularly botnets.

We start the paper with an introduction to botnets in section 2. Section 3 describes the underlying building blocks of our approach by detailing the host dependency model and data mining techniques. Comprehensive experiments are discussed in section 4. We address related work in section 5 and conclude the paper in section 6.

2 Structured Botnets

2.1 Overview

A botnet is a network of compromised hosts usually called bots or zombies. The botnet owner, also known as the botmaster or controller manages the bots. Currently, botnets make themselves known via the threats they pose as for example: highly distributed denial of service attacks, data stealing or large spam campaign. By not involving all available bots, spam [38] can be generated more stealthily in order to avoid detection. Another feature of bots is the permanent multiplication effort: bots attempt to scan and infect other machines and so grow the network. Attacks are strongly also linked to financial interests [13].

2.2 Botnet Architectures

The botmaster manages her bots through a Command and Control (C&C) channel. In the past, this was generally a pseudo-centralized architecture based IRC [26] (Internet Relay Chat) channel [24]. Since distributed architectures are more scalable and robust, attackers also chose this kind of architecture for constructing botnets. Slapper [3] was the first botnet worm in this category, but structured peer-to-peer (P2P) protocols followed due to their underlying scalability and robust communication facilities [10]. In structured P2P architecture, a huge ID space is defined, a typical value being $N = 2^{160}$. Each host (corresponding to an IP address and a UDP port) has a node ID in this space and a routing table corresponding to hosts that can be directly reached. Each node is also responsible for data information (files) having a key (hashed value) close to its own. For other nodes, a routing process is employed and the main advantage of structured P2P is that reaching a node is guaranteed in a maximal number of

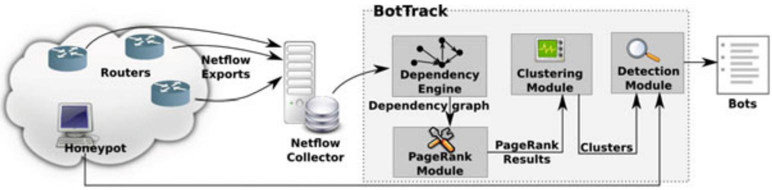


Fig. 1. BotTrack architecture

hops. Chord [33] was one of the pioneering works and guarantees a routing in $log(N)$ hops. Koorde [17] is an optimization of Chord which provides a routing complexity equal to $O(log(N)/log(log(N))$. Storm/Peacomm[28] is a infamous real worm using both cryptographic protection and peer-to-peer communication models [14]. Storm is based on the Kademelia [23] protocol usually used for file sharing. Storm bots look for specific keys within the P2P network corresponding to systems where the botmaster provides necessary information for connecting to a webserver publishing orders to execute. The Kademelia protocol has a searching algorithm that finds these hosts in $O(log(N))$. We argue in this paper that analyzing communication patterns in network traffic can reveal botnets relying on such communication models. By studying Chord which can be seen as the generic protocol for DHT (Distributed Hash Tables) and some extensions in different directions (Kademlia with a high redundancy and Koorde with a very small one), it is enough representative of most of P2P botnets.

3 Bot Detection

3.1 Architecture

Figure 1 highlights the main components of our approach. Routers monitor the traffic and export NetFlow records to a collector. BotTrack then analyzes this data. The first step identifies the interactions between systems by creating a dependency graph between hosts. This graph is then automatically analyzed by a module running the PageRank algorithm [27] to extract the nodes (IP addresses) which are strongly linked to each other. These linkage levels provide authority and hub scores on which a clustering algorithm is applied to find nodes with similar roles within the network. However, nodes playing the same role may not necessarily be P2P nodes or P2P bot nodes. For more precision, data from a honeypot is also leveraged. Hence, some nodes are known a priori to be bots and clusters including such points are considered to be groups of bots to which mitigation techniques can be applied.

3.2 Flow Monitoring

IP flow [29] is a standard tool for today's large-scale network monitoring.

A flow is a series of IP packets sharing the same source, destination address, associated ports, and protocols. For high speed networks, flow monitoring is well used because, firstly, network traffic analysis via flow records is much more efficient than via individual packets (less data to analyze), and secondly, flow records require only a fraction of the storage space needed for full packet capture. The flow collecting architecture consists of probes and collectors. Probes or sensors are devices deployed in different network locations, and are responsible for capturing flow data and forwarding it to the collector. Almost all modern commercial-grade routers support flow record export. In this paper, we use without any distinction the terms IP flow [29] and NetFlow [7] which reflects different standards.

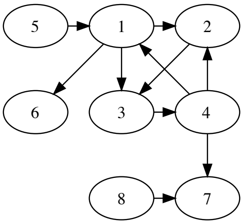


Fig. 2. Sample host communication graph example (trivial example bots on nodes 1,2,3 and 4)

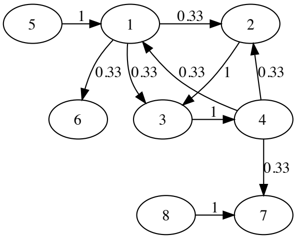


Fig. 3. simplified ranking distribution after 1st iteration

3.3 Host Communication Model

Because of its wide availability and intrinsic host communication information we utilize flow data and link analysis to detect structured botnets. The rationale behind our approach is that P2P structured botnets exhibit a distinguishable communication pattern among bots. Our communication dependency graph engine (see figure 1) generates a directed "who talks to whom" host communication graph. Figure 2 shows a trivial example of a host communication graph of 8 nodes. Each node represents a host, and a directed link from node A to B indicates there is at least one IP flow from host A to host B. It is obvious that nodes 1, 2, 3, and 4, which represent bot nodes, exhibit more intense communications among themselves than with other nodes. The following sections will explain how to leverage the PageRank link analysis algorithm bounded by clustering techniques to detect bots via host communication graphs. The terms host communication graph and dependency graph are used interchangeably in this paper.

3.4 PageRank

The PageRank algorithm [27] is a link analysis algorithm used by the Google web search engine to weight the relative importance of web pages on the Internet. PageRank ranks each web page according to a a score depending on the hyperlink structure of the Internet. A web page is referred to by other pages if other pages contain a hyperlink pointing to it. Intuitively, a frequently-referred page is important, and pages referred to by few important pages are also considered important. For instance, if a web page is linked by only one web site, but this web site is Yahoo, then this page should be considered as important as well. The outstanding scalability and efficiency of PageRank makes it an ideal candidate to analyze the link structures among communicating hosts on large scale networks.

PageRank is computed in an iterative fashion. At each iteration, the current score of each node is distributed through its outgoing links to the nodes it points to. The new rank of each node is the sum of the scores its incoming edges bring. For example, assuming an initial value of each node in Figure 2 of 1, the ranking distribution following the first iteration is shown in Figure 3. Node 1 has three outgoing links, thus each outgoing link takes $\frac{1}{3}$ ranking to its destination node. Node 1 receives two incoming links from node 4 and node 5, thus the new score of node 1 is $\frac{1}{3} + \frac{1}{1} = 1.33$. The iterations continue until the score changes are sufficiently small or the maximum number of iterations is reached.

Based on some knowledge, important nodes scores can be increased by prior defined weight. In the botnet context, it corresponds to nodes that are known to be bots and should have a higher impact on the score calculation and propagation.

Let $P_t(i)$ denote the PageRank score of node i for iteration t, (j, i) denotes a directed edge from node j to node i, O_j represents outgoing link number of node j, and E is the set of directed edges on the whole graph. Based on some knowledge, a subjective preference can be assigned to certain nodes. This is useful in the context of bot detection in the sense that we can promote the rankings of hosts known to be bots. The subjective influence functions via the W component and the impact of this weight and the propagated scores can be adjusted by tuning the d parameter:

$$P_t(i) = (1 - d) \sum_{k=1}^{n} W(k) + d \sum_{(j,i) \in E} \frac{P_{t-1}(j)}{O_j} \tag{1}$$

For flow monitoring, it is interesting to consider both edge directions. When edges point from source to destination hosts, the ranking scores distribute towards the final destination of traffic, and is named hub rank in this paper; when edges point from destination to source hosts, the ranking scores distribute towards the root cause, and we name this root-cause-selecting ranking authority rank. For the latter, PageRank algorithm is executed again with the graph where the edge directions have been inverted. Obviously, the PageRank algorithm can be seen as a set of matrix computations and theoretical details may be found in [27].

3.5 Clustering

PageRank output is composed of two values: authority and hub. Both are useful for detecting P2P bots since peers in such network have to initiate connections with many others and are also destinations of many connections. A simple approach should consider as a P2P bot a host associated with a high ranks for these two metrics:

$$authority > \theta \ and \ hub > \gamma \tag{2}$$

Although defining two thresholds is not easy, section 4 shows that neither can be excluded since good results are obtained only when both are used. Furthermore, infected machines can be clustered in disjoint clusters that are not necessarily adjacent in terms of their traffic flows. This is illustrated in figure 4 by showing the distribution of ingoing and outgoing node degrees. In brief, defining only a simple threshold is not sufficient and investigators should define different ranges of authority and hub values for which the hosts should be considered as bots. Finally, defining such ranges is often dependent of the P2P protocol used (see

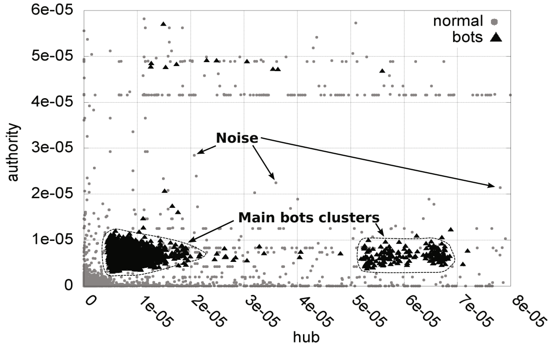


Fig. 4. Kademelia authority and hub values for normal hosts and bot hosts

section 4). For example, figure 4 clearly highlights such ranges for bots, although a minority of normal hosts may have higher values, even though the majority of normal hosts is associated with very low values.

Therefore, applying a clustering technique may be helpful, and limits the number of parameters that must be chosen. Different techniques could be applied but the following criteria have to be considered:

1. few parameters to set
2. no knowledge about the number of clusters, since different kinds of machines could be infected
3. clusters can have irregular shapes because normal communications are also considered
4. since non-infected hosts are also diverse, authority and hub values can be very different.

Several algorithms fulfill these requirements and our method is based on DBSCAN [4] because only two parameters have to be tuned, and the complexity of this algorithm is acceptable especially when using specific data indexation [4]. In our case, the runtime is slightly above linear. Moreover, there are many extensions for speeding up the clustering process [22] and for using parallel computing [39]. All these works show that DBSCAN is quite fitted for huge databases like usual NetFlow database of real networks.

DBSCAN is a density-based algorithm where the key idea is to consider that each point of a cluster has to be enough close to a minimal number of points, $MinPts$, which represents the neighborhood defined by a maximal distance ϵ. Such a point is qualified as a core point and all points whose the distance from it is lower than ϵ are also included in the cluster and are qualified as a border point if they are not core points. Thus, points which are isolated (neither core nor border) are considered to be noise. More precisely, the algorithm iterates over every point p and does the following for each point that is not yet classified:

1. compute the neighbors: $neighbor = \{p2, dist(p, p2) < \epsilon\}$. In this paper, we use the Euclidian distance.
2. if $|neighbor| \geq MinPts$, create a new cluster containing p and all points of $neighbor$.
3. go back to step 1 by taking previous neighbors as the original node to compute the neighbors and by excluding previously assigned points.

Thus, DBSCAN has only two parameters to tune but we assume $MinPts = 4$ since botnets can sometimes contain only hundreds of hosts and so clusters of bots can be relatively small. However we expect few of them to share similar hub and authority values. Thus, the main parameter to tune in our experiments was ϵ.

4 Experimental Result

4.1 Methodology and Datasets

Our evaluation was made on a real dataset provided by a major Luxembourg Internet operator. We extracted 13.7GB of data described in table 1. Following the approach of [25], we added synthetic botnet flow records without taking into account the duration or the size of flows since our approach does not rely on them. This is the only viable solution since getting a labeled dataset at an operator level is not possible. Hence, the corresponding characteristics are always the same in table 1. To evaluate the robustness of our system with a stealthy botnet, we extracted only 1% of IP addresses for constructing the botnet. As we relies exclusively on IP addresses, other features like timing or the number of bytes are randomly generated for the botnet flows. So, the attack scenario may be simply explained as adding bot to bot communications at the end of the capture by examining routing table entries of each bots.

Moreover, we assume a random distribution of bot IP addresses. By this way, the IP addresses selected as bots can be considered indifferently as the whole botnet or as a part of bigger botnet containing IP addresses also outside form the monitored network. Hence, there is no need to define different scenarios when having a global view of the botnet is unavailable.

Kademlia topology entails many links between hosts since by design it aims to improve performance for file sharing based on multiple paths. Therefore, this topology exhibits the highest number of flows in table 1. Chord is a more compact topology (no multiple paths from a node to a space partition) which implies lower performance. Neighbor node in Koorde ID space share links in order to contact far-distant nodes. Thus each individual node has only to maintain a few links and the number of flows is reduced.

Table 1 also presents general statistics about PageRank results. Just 1% of bots does not impact highly on average values, but the extreme values indicate that there are some changes in the graph topology.

Our evaluation is based mainly on the Receiver Operating Characteristic (ROC) [9] which is well-suited for measuring the efficiency of two classes of classification. The curve measures the True Positive Rate (TPR) against the

Table 1. Flow record statistics (original data + botnet traces)

	chord	Kademlia	Koorde
Flow#	2133887	2399032	1997049
Host#	323610	323610	323610
Bytes#	13.7G	13.7G	13.7G
Duration	18min23sec	18min23sec	18min23sec
Max HR	0.0472	0.0436	0.0442
Min HR	1.264e-08	1.185e-08	1.185e-08
Avg HR	3.0901e-6	3.0902e-06	3.0902e-06
Max AR	0.022897	0.022534	0.023161
Min AR	1.3229e-08	1.1429e-08	1.19e-08
Avg AR	3.090145e-06	3.090138e-06	3.0901e-06

False Positive Rate (FPR). The TPR is the number of bots correctly identified as bots (also called True Positive or TP) divided by the total number of bots. The FPR is the number normal hosts identified as bots (also called False Positive or FP) divided by the total number of benign hosts.

4.2 Pure Link Analysis

Figure 5 illustrates the simple threshold-based method to detect bot links in equation (2). However, in order to measure the efficiency of both metrics (authority and hub), these are considered separately. Hosts with a value higher than a threshold are considered to be bots. Figure 5 displays the ROC curve when the threshold varies. The results are good, particularly for hub detection. This suggests the server role of P2P hosts is more discriminative than the client role. Due to the different linkage level of topologies and so on in the volume of flow records highlighted in table 1, the Kademlia based botnet is the easiest to discover, and Koorde is the best topology for avoiding detection.

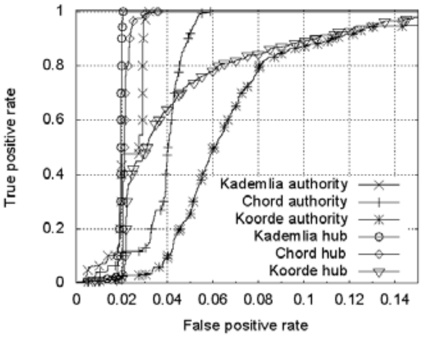
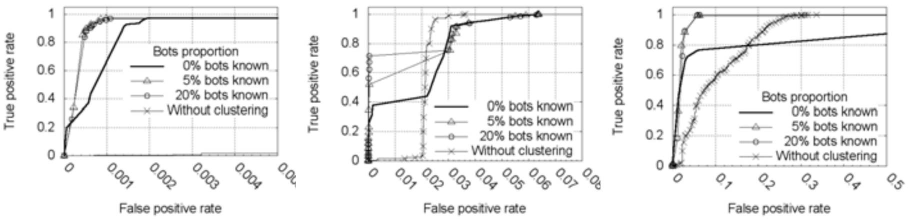


Fig. 5. ROC curves without clustering



(a) Kademlia **(b)** Chord **(c)** Koorde

Fig. 6. Clustering impact (without clustering is the results of the simple method based on a threshold and the hub values)

Even if the number of false positives is quite low in most of cases, 0.02 was equivalent to more than 6400 FPs. Thus, taking steps to discard them is essential to avoid blocking too many benign hosts.

4.3 Clustering

In this experiment, clustering was applied to the link analysis results to improve the results by creating clusters of IP addresses. However, once the clusters have been created, it is not clear how to decide which one is a botnet. Hence, we consider that if a cluster contains at least one bot, all points included within it are also bots. This means that we need to know at least one bot for each cluster of bots. This knowledge can be obtained by using intrusion detection systems or honeypots. Meanwhile, it may lead to many false positives but the results described below show that botnet clusters are quite consistent. The difficulty of this task clearly depends on the number and the structure of clusters. Figure 6 highlights good results when applying clustering to the different topologies (curves entitled "0% bots known"). For Kademelia in figure 6a, the results are greatly improved (the original curve is very close to 0 in figure). This means that clustering allows also allows a high TPR to be reached with fewer FP. The same observations can be made for other topologies (figures 6b and 6c) but the only difference is that after a certain threshold of false positive rate, the true positive rate is better without clustering. This is due to bot points which are less distinguishable with such topologies and which are not clustered, because these are considered as noise.

Finally, we consider the inflection point as a good configuration for reducing false positives. It corresponds to the values in table 2, which highlights a large reduction of false positives compared to the threshold-based method, in particular for Kademelia and Chord with about 90% of false positives discarded. Table 2 also highlights that the setting of the parameter ϵ is clearly dependent on the topology. Hence, including a new topology is not straightforward and a priori parameter estimation is needed.

The various performance measures are quite acceptable, except in the case of Chord. From a general point of view, clustering is able to eliminate many false positives, especially in the left part of the curves. However, the results are

Table 2. Clustering impact in best cases (inflexion point)

Topology	ϵ	TPR	FPR	FP reduction	#clusters
Kademlia	2.5e-05	0.97	0.0019	5945 (90.1%)	22
Chord	4e-07	0.38	0.002	6087 (90.3%)	13
Koorde	1e-04	0.76	0.06	32037 (59.1%)	227

still acceptable in the right hand part, even if lower than without clustering. However, sometimes it could be better to detect only a subset of a botnet in order to avoid blacklisting too many benign hosts at the same time.

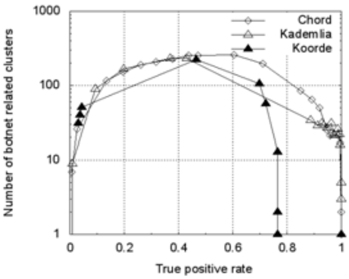
4.4 Impact of the Number of Known Bots

In this experiment, we assumed that some individual bots had already been detected by other programs such as a honeypots. We influenced PageRank algorithm by such knowledge by personalizing the initial weights on nodes. Figure 6 shows the results when 5 or 20 percent of bots are known. The results are improved significantly. Hence, Chord botnet-based detection is more acceptable when applying clustering with such knowledge. Considering Koorde and Kademlia, clustering always provides the best results as it was also the case without bot knowledge. For instance, detecting around 99% of bots is possible with a FPR of only 0.001. Compared to 0.002 (the FPR without bots known), this means that FP, were halved, representing 320 FPs discarded.

4.5 Cluster Analysis

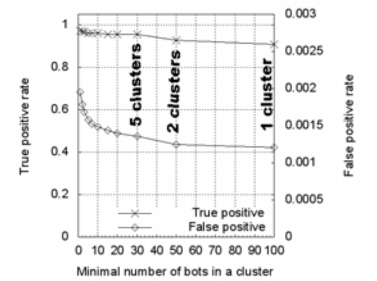
In our previous experiments, we considered that we were able to identify one bot per created cluster by deploying a probe, such as a honeypot. It can be very easy to obtain good results just by creating small clusters with a minimal number of points. The extreme case is one point per cluster. Such points are considered as noise by DBSCAN and will not be considered further. Figure 7 shows that a high TPR is possible with few clusters. Having too many clusters does not produce the best results, and the TPR increases considerably when the number of clusters is reduced. Obviously, the reader has to refer to figure 6 for the FPR since it is possible to detect all bots with only one big cluster, but this implies a high FPR. For instance, we can consider the configurations in table 2, showing that for Kademlia and Koorde, the number of bots clusters is very limited. Hence, in Kademlia, 22 bots must be known. The number of clusters in Chord is quite high for this particular configuration. Assuming the second inflection point in figure 6 (TPR = 0.96, FPR = 0.04), the number of clusters is reduced to 21.

Unfortunately knowing around 20 bots from different clusters is not always possible. This occurs when some clusters are very small and only contain a limited number of bots. However, in this case, discarding them has a limited impact on TPR. The final experiments assume that clusters with only a minimal number of bots can be detected. As highlighted in figure 8, the TPR is not impacted positively since it slightly decreases. For instance, considering that we



[t]

Fig. 7. Number of bot clusters



[t]

Fig. 8. Kademlia – Small cluster exclusion

have clusters with at least 50 bots yields only 2 clusters while keeping a TPR of 92%. This case correspond to only discovering the main clusters. (See figure 4). Besides, the FPR is more greatly affected and decreases, particularly in the early phases.

5 Related Work

Monitoring of IP flow records was first addressed in [32] by providing statistics on traffic volume and underlying IP addresses in order to apply time-series based anomaly detection. Follow-up works [31] employs Hidden Markov Models (HMM) for traffic modeling. The obvious way to observe a botnet is to deploy a honeypot in order to be infected. This is very efficient with IRC botnets for tracking the servers based on the IP addresses and the DNS (Domain Name System) names [1]. Many detection methods rely on first detecting malicious activities such as spam or scanning activities [20]. Some approaches aim to correlate these malicious activities with C&C detected communications [12,11]. Crawling a P2P botnet is an alternative way to detect it [13]. When the P2P botnet employs a normal P2P network, abnormal ID or IP addresses variation can be used as discriminating features [34]. Connecting components of a graph representing host interactions is presented in [8] and analysis methods can be found in [15]. Building service-level or host-level dependencies has been addressed in the past [6,18,30,2,16] for supporting network management tasks. BLINC [19] also leverages inter-host communication relationships for traffic classification, which is also well-studied in [21]. The closest related work is [25], which is based on random walk and clustering techniques to isolate interaction subgraphs related to P2P communications. Our previous work has leveraged link analysis algorithms on flow dependency graphs [35,36] and host dependency graphs [37] for tracing the root cause flow records corresponding to the malicious traffic. They help to select PageRank as the much suitable algorithm for this paper. Flow dependency graphs [35,36] do not consider hosts as nodes but flows as nodes and try to build causal connections between them based on timing analysis. Besides, our current work differs from the previous works in that we enhanced the link analysis result with clustering techniques to reveal the bots on large-scale networks.

6 Conclusion and Future Work

In this paper we have addressed the automated detection of P2P based botnets. Our approach is based on processing NetFlow-related information to build a host dependency model that captures information about which host is talking to which other host. Linkage analysis on this host dependency model is coupled with a clustering algorithm in order to build clusters of similarly behaving nodes. We show in this paper that bot-infected systems tend to belong to the same cluster. Additional information sources, such as honeypots, can be also used for tagging specific clusters. This will help to distinguish benign P2P applications from botnets. We have shown the viability of our method on large datasets from a major Internet service provider in Luxembourg. Because no labeled datasets exist in the research community, we had to follow a similar approach to [25] and inject synthetic data into the ISP-sourced datasets. We plan to do a complete complexity studies about our solutions for defining optimizations since the current implementation on a single machine handles 18min of Netflow records in around 160 seconds.

Acknowledgment. The authors would like to thank S. Nagaraja for discussions and advice on our work, especially on how to generate realistic datasets [25].

References

1. Abu Rajab, M., Zarfoss, J., Monrose, F., Terzis, A.: A multifaceted approach to understanding the botnet phenomenon. In: ACM SIGCOMM Conference on Internet Measurement (IMC), pp. 41–52 (2006)
2. Aguilera, M., Mogul, J., Wiener, J., Reynolds, P., Muthitacharoen, A.: Performance debugging for distributed systems of black boxes. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, pp. 74–89 (2003)
3. Arce, I., Levy, E.: An analysis of the slapper worm. IEEE Security and Privacy 1(1), 82–87 (2003)
4. Berkhin, P.: A survey of clustering data mining techniques. In: Grouping Multidimensional Data, pp. 25–71. Springer, Heidelberg (2006)
5. Buxbaum, P.: The fog of cyberwar – to defend... and attack?, http://www.isn.ethz.ch/isn/Current-Affairs/Special-Reports/The-Fog-of-Cyberwar/Botnets/ (accessed on 08/30/10)
6. Chen, X., Zhang, M., Mao, Z.M., Bahl, P.: Automating network application dependency discovery: Experiences, limitations, and new solutions. In: Proceedings of OSDI (2008)
7. Claise, B.: Cisco Systems NetFlow Services Export Version 9. RFC 3954 (Informational) (October 2004)
8. Collins, M.P., Reiter, M.K.: Hit-list worm detection and bot identification in large networks using protocol graphs. In: Kruegel, C., Lippmann, R., Clark, A. (eds.) RAID 2007. LNCS, vol. 4637, pp. 276–295. Springer, Heidelberg (2007)
9. Fawcett, T.: An introduction to roc analysis. Pattern Recogn. Lett. 27(8), 861–874 (2006)

10. François, J., State, R., Festor, O.: Towards malware inspired management frameworks. In: IEEE/IFIP Network Operations and Management Symposium (NOMS), pp. 105–112 (2008)
11. Gu, G., Perdisci, R., Zhang, J., Lee, W.: Botminer: clustering analysis of network traffic for protocol- and structure-independent botnet detection. In: USENIX Security Symposium (SS), July 2008, pp. 139–154. San Jose, CA (2008)
12. Gu, G., Porras, P., Yegneswaran, V., Fong, M., Lee, W.: Bothunter: detecting malware infection through ids-driven dialog correlation. In: USENIX Security Symposium (SS) (August 2007)
13. Holz, T., Steiner, M., Dahl, F., Biersack, E., Freiling, F.: Measurements and mitigation of peer-to-peer-based botnets: a case study on storm worm. In: Workshop on Large-Scale Exploits and Emergent Threats (LEET). USENIX (2008)
14. Hund, R., Hamann, M., Holz, T.: Towards next-generation botnets. In: EC2ND: European Conference on Computer Network Defense, pp. 33–40. IEEE Computer Society, Los Alamitos (2008)
15. Iliofotou, M., Faloutsos, M., Mitzenmacher, M.: Exploiting dynamicity in graph-based traffic analysis: techniques and applications. In: ACM International Conference on Emerging Networking Experiments and Technologies, CoNEXT (2009)
16. Jian-Guang, L., Qiang, F., Wang, J.Y.: Mining dependency in distributed systems through unstructured logs analysis. research.microsoft.com `http://research.microsoft.com/pubs/101994/Dependency%252520Camera%Ready.pdf`
17. Kaashoek, M.F., Karger, D.R.: Koorde: A simple degree-optimal distributed hash table. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735. Springer, Heidelberg (2003)
18. Kandula, S., Chandra, R., Katabi, D.: What's going on?: learning communication rules in edge networks. In: Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication, pp. 87–98 (2008)
19. Karagiannis, T., Papagiannaki, K., Faloutsos, M.: BLINC: multilevel traffic classification in the dark. In: ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM (2005)
20. Karasaridis, A., Rexroad, B., Hoeflin, D.: Wide-scale botnet detection and characterization. In: First Workshop on Hot Topics in Understanding Botnets (HotBots). USENIX (2007)
21. Kim, H., Claffy, K., Fomenkov, M., Barman, D., Faloutsos, M., Lee, K.: Internet traffic classification demystified: myths, caveats, and the best practices. In: ACM CoNEXT (2008)
22. Kryszkiewicz, M., Skonieczny, Ł.: Faster clustering with dbscan. Intelligent Information Processing and Web Mining, pp. 605–614 (2005)
23. Maymounkov, P., Mazières, D.: Kademlia: A peer-to-peer information system based on the XOR metric. In: IPTPS 2001: International Workshop on Peer-to-Peer Systems, pp. 53–65. Springer, Heidelberg (2002)
24. McLaughlin, L.: Bot software spreads, causes new worries. IEEE Distributed Systems Online 5(6) (2004)
25. Nagaraja, S., Mittal, P., Hong, C., Caesar, M., Borisov, N.: BotGrep: Finding p2p bots with structured graph analysis. In: Security Symposium. USENIX (2010)
26. Oikarinen, J., Reed, D.: rfc 1459: Internet relay chat protocol (1993)
27. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web (1998)
28. Porras, P., Sadi, H., Yegneswaran, V.: A Multi-perspective Analysis of the Storm (Peacomm) Worm, `http://www.cyber-ta.org/pubs/StormWorm/SRITechnical-Report-10-01-Storm-Analysis.pdf`

29. Quittek, J., Zseby, T., Claise, B., Zander, S.: Requirements for IP Flow Information Export (IPFIX) (2004), `http://www.ietf.org/rfc/rfc3917.txt`
30. Reynolds, P., Wiener, J.L., Mogul, J.C., Aguilera, M.K., Vahdat, A.: Wap5: blackbox performance debugging for wide-area systems. In: Proceedings of the 15th International Conference on World Wide Web, pp. 347–356 (2006)
31. Sperotto, A., Sadre, R., de Boer, P., Pras, A.: Hidden markov model modeling of ssh brute-force attacks. In: Integrated Management of Systems, Services, Processes and People in IT, pp. 164–176.
32. Sperotto, A., Sadre, R., Pras, A.: Anomaly characterization in flow-based traffic time series. IP Operations and Management, 15–27
33. Stoica, I., Morris, R., Karger, D., Kaashoek, F., Balakrishnan, H.: Chord: A scalable Peer-To-Peer lookup service for internet applications. In: Proceedings of the 2001 ACM SIGCOMM Conference, pp. 149–160 (2001)
34. Wang, B., Li, Z., Tu, H., Hu, Z., Hu, J.: Actively measuring bots in peer-to-peer networks. In: Networks Security, Wireless Communications and Trusted Computing (NSWCTC). IEEE, Wuhan (2009)
35. Wang, S., State, R., Ourdane, M., Engel, T.: FlowRank: Ranking netflow records. In: Proceedings of the 6th International Wireless Communications and Mobile Computing Conference (2010)
36. Wang, S., State, R., Ourdane, M., Engel, T.: Mining netFlow records for critical network activities. In: Stiller, B., De Turck, F. (eds.) AIMS 2010. LNCS, vol. 6155, pp. 135–146. Springer, Heidelberg (2010)
37. Wang, S., State, R., Ourdane, M., Engel, T.: Riskrank: Security risk ranking for ip flow records. In: Proceedings of the 6th International Conference on Network and Services Management, CNSM 2010 (2010) (to appear)
38. Xie, Y., Yu, F., Achan, K., Panigrahy, R., Hulten, G., Osipkov, I.: Spamming botnets: signatures and characteristics. SIGCOMM Comput. Commun. Rev. 38(4), 171–182 (2008)
39. Xu, X., Jäger, J., Kriegel, H.P.: A fast parallel clustering algorithm for large spatial databases. Data Min. Knowl. Discov. 3(3), 263–290 (1999)