Towards Automated Detection of Peer-to-Peer Botnets: On the Limits of Local Approaches

Márk Jelasity
University of Szeged, Hungary, and
Hungarian Academy of Sciences

Vilmos Bilicki
University of Szeged, Hungary

Abstract

State-of-the-art approaches for the detection of peer-to-peer (P2P) botnets are on the one hand mostly local and on the other hand tailored to specific botnets involving a great amount of human time, effort, skill and creativity. Enhancing or even replacing this labor-intensive process with automated and, if possible, local network monitoring tools is clearly extremely desirable. To investigate the feasibility of automated and local monitoring, we present an experimental analysis of the traffic dispersion graph (TDG)—a key concept in P2P network detection—of P2P overlay maintenance and search traffic as seen at a single AS. We focus on a feasible scenario where an imaginary P2P botnet uses some basic P2P techniques to hide its overlay network. The simulations are carried out on an AS-level model of the Internet. We show that the visibility of P2P botnet traffic at any single AS (let alone a single router) can be very limited. While we strongly believe that the *automated* detection and mapping of complete P2P botnets is possible, our results imply that it cannot be achieved by a *local* approach: it will inevitably require very close cooperation among many different administrative domains and it will require state-of-the-art P2P algorithms as well.

1 Introduction

P2P botnets are a challenge to the security of the Internet that is rather difficult to underestimate. Considering the fact that P2P botnets have not even begun to fully utilize the increasingly advanced P2P techniques for their advantage, the future seems even more challenging.

State-of-the-art approaches for detecting P2P botnets rely on considerable human effort: a specimen of the P2P bot needs to be captured and reverse engineered, message exchange patterns and signatures need to be extracted, the network needs to be crawled using tailor-made software, its indexing mechanism poisoned, or otherwise infiltrated, and a countless number of creative techniques has to be applied such as visualizations, identifying abnormal patterns in DNS or blacklist lookups, understanding propagation mechanisms, and so on (e.g., [3, 5, 23]). Besides, aggressive infiltration and crawling introduces lateral damage: it changes the measured object itself very significantly [12].

While creative and knowledge-intensive approaches are obviously useful, it would be important to be able to detect and map botnets *automatically*, and as *generically* as possible. Ideally, network monitoring and filtering tools installed at routers and other network components should take care of the job with very little human intervention based on the (often enormous volumes of) Internet traffic constantly flowing through them.

This automation problem has been addressed in the context of IRC-based botnets [26] and, recently, also in the context of detecting generic botnet activity [4] and, specifically, P2P traffic [8].

In this paper we examine how techniques presented in [8] perform in the case of botnets. Instead of looking at real traffic traces, we base our methodology on simulation: we define synthetic flows on top of an AS-level model of the Internet assuming various P2P botnets. This is necessary because our main goal is not to evaluate current botnets, but rather to explore some advanced P2P techniques such as localization and clustering that *future* P2P botnets are likely to adopt to avoid detection.

Our main contribution is demonstrating that P2P botnets can easily hide their traffic even at the largest backbone router if they apply a few P2P techniques that are available from the literature or that are fairly evident, while being able to maintain their overlay and hence their malicious activity as well.

The implication is that local and isolated efforts are not very promising; if we would like to protect the Internet from P2P botnets that are likely to increase in size and sophistication, and that are still very far from their full potential, we need to fight fire with fire and start to devote serious efforts to the consideration of P2P infrastructures and algorithms for automated detection.

2 Focusing on Overlay-Related Traffic

If we want to identify and filter P2P botnet traffic in an automated way, we can focus on roughly three kinds of activity: propagation, attacks by the botnet, and overlay traffic, which involves repairing failed overlay links, adding new nodes to the overlay, and spreading and searching commands of the botmaster.

We argue that the most promising approach is to focus on overlay traffic. It has a small volume, but it is arguably the most regular and reliable traffic any P2P botnet generates, since the overlay network has to be constantly repaired, and bots need regular information about commands of the botmaster, updates, and so on.

Although the propagation of bots can be detected if it involves port scanning and similar suspicious network activity (e.g., [1]) bots can also spread under the radar via email, websites, file sharing networks, ad hoc wireless networks, or even via the old-fashioned way by infecting files on pen drives and on other portable media that generate no network traffic at all [15, 27].

Certain "visible" attacks—such as DDos, spamming or brute force login—could also be detected automatically. In an optimistic scenario, we can immediately identify and block those bots that participate in these attacks. But this is still far from enough: P2P overlays can apply very cheap and simple methods that can enable them to tolerate extremely well if large portions (even 80%) of the overlay gets knocked out [10]. In addition, certain types of malicious activity, such as collecting personal data (bank account information, passwords, etc) can blend into (even piggyback) overlay traffic perfectly.

One interesting idea is that—instead of concentrating just on the overlay or only on attacks—we should look for a *correlation* between groups of nodes that produce a similar flow-level behavior due to overlay traffic and groups that perform attacks, as proposed in [4]. However, as acknowledged in [4], such correlations could be reduced to a minimum by a sophisticated botnet.

The basic motivation of our work is that we think that the *structure* of P2P networks is a very promising, although difficult, target to try to detect. The structure is completely insensitive to actual flow characteristics: Nodes can mimic other protocols or they can behave randomly, but the traffic dispersion graph they generate must still reveal the overlay network they are organized in. Evidently, this structure will have to be correlated to malicious behavior as well.

Accordingly, in the rest of the paper we focus on overlay traffic.

3 Network Monitoring with Traffic Dispersion Graphs

Approaches to automated traffic classification and filtering typically start from observing packets or flows and classify them through the application of a stack of methods ranging from simple port-based filtering to sophisticated supervised or unsupervised machine learning and classification methods over packet and flow data. A summary of such methods is given in [22].

However, P2P botnet overlay traffic does not necessarily look malicious or harmful (in fact in itself it is neither), even if isolated and classified properly. It is essential to be able to identify this traffic as part of a *network* that, as such, makes it suspicious and could trigger a warning [2].

It has already been argued that it is very difficult to detect P2P traffic using packet or flow classification methods alone [7, 8]. Most of the key characteristics of P2P traffic lie in the network defined by the flows, the so-called traffic dispersion graph (TDG). By building and analyzing TDGs of locally observable flow data after the classification phase, it is possible to extract important additional clues about the organization of an application and, for example, label them as P2P.

In [7], the TDG is defined on top of a set of flows S that have the usual format <srcIP, srcPort, dstIP, dstPort, protocol>. The TDG is the directed graph $G(V, E)$ where V, the set of vertices, contains the set of IPs in S, and E, the set of edges, contains the edges (a, b) such that there is a flow in S with srcIP $= a$ and dstIP $= b$.

Since the present work intends to challenge the feasibility of local methods for detecting botnets, in order to be convincing we need to be generous to the local methods that are available for traffic classification. Therefore we assume that *traffic that belongs to a given P2P botnet can be isolated in an unlabeled way*. That is, we assume that there are methods available to group a set of flows together that belong to the P2P botnet, but that we cannot determine whether the identified class of traffic is in fact botnet traffic.

Note that this is already an extremely strong assumption. We will argue that even with this assumption, it is very difficult to identify the traffic as P2P traffic generated by a large P2P network if a P2P botnet applies certain P2P techniques. To show this, we examine how the P2P traffic identification approach presented in [8] performs in this context.

4 Our P2P Overlay Model

There is a huge number of design options for P2P overlays [16], so selecting a suitable model that allows us to draw conclusions on the detectability of P2P botnet overlay traffic in general is highly non-trivial. First we give a very brief bird's-eye-view of existing overlay designs, and based on this overview we propose a simple model. Finally, we present techniques that could help a P2P botnet to hide; we will examine these techniques in our experiments in Section 5.

4.1 P2P Overlays

Unstructured P2P Overlays The broad class of *unstructured* overlays refers to random topologies with different degree distributions such as power-law networks [24] or uniform random networks [10]. They offer no possibility for routing or key lookup, and they support flooding, random walk, or variations of these, as search methods. Gossip protocols are also well supported [13].

Superpeer Overlays In superpeer networks peers are not equal: a small subset of the peers are automatically selected as temporary servers to help functions such as search and control [21]. Many P2P applications such as Skype, FastTrack and Gnutella [16] apply superpeers. Since superpeer networks are more visible, and less robust to targeted attacks, we assume that the most efficient botnets are not likely to adopt this design.

Structured P2P Overlays Structured overlays are distributed linked data structures designed for efficient routing (ID search). Nodes have unique IDs that can be used to address them. The overlay is organized to make efficient routing to a given ID possible. All versions of these overlays can be described as having a local structure based on some metric (often a ring, along which the IDs are ordered) and long range links that serve as shortcuts. Shortcuts are typically arranged in a way that is consistent with the optimal arrangement described in [14]. Current P2P botnets are based on structured overlays such as Kademlia [3].

4.2 Our Model

Unstructured networks are extremely robust, and, due to their lack of structure, they can be harder to discover as well. However, command and control operations are more expensive; and, most importantly, communication cannot be localized (an important technique we describe below) since we have no structure to map on the underlay. Although we do not rule out unstructured networks as a potential architecture for botnets, here we focus on structured networks.

As a model we use an ordered ring with exponential long range links, a simplified version of the Chord topology [25]: We have N nodes with IDs $0, 1, \dots, N-1$. Node i is connected to nodes $i - 1 \pmod N$ and $i + 1 \pmod N$ to form the ring. In addition, node i is connected to nodes $i + 2^j \pmod N$ for $j = 1, 2, \dots, (\log_2 N) - 1$, which are the long range links.

It is important to make a distinction between the overlay and the flows that exist in the overlay. Two nodes a and b are connected in the overlay if a "knows about" b. This, however, does not imply that a will ever actually send a message to b. For example, a might remember b simply in order to increase robustness in case of a failure. On the other hand, a node a might send a message to b even though b is not the neighbor of a in the overlay (that is, for the overlay to function properly, a does not need to remember b after sending it a message). For example, in Kademlia if a wants to find the node of ID x then a will actually contact all nodes on the route to x. This is why Storm bots generate so many messages locally as part of the overlay traffic [5].

In short, we want to model the flows and not the overlay *per se*, so our model refers to the *flows* we can potentially observe. In the actual overlay there would probably be links to the 2nd, 3rd, etc, neighbors in the ring as well that are learned from direct neighbors.

In the following we describe two fairly straightforward techniques that future botnets could use to hide their traffic. The key point is that, using these techniques, the functionality of the overlay can be preserved while using far fewer links and traversing fewer routers.

4.2.1 Clusters for Sharing Long Range Links

In the ring every node has two neighbors that it actually communicates with at any given time, but it has $\log N$ long range links all of which are frequently used for communication to achieve as few as $O(\log N)$ hops in overlay routing (where N is the network size). Evidently, the ring would be sufficient for communication but then sending a message from a node to another random node would require $O(N)$ hops in expectation.

There is a middle ground: we can reduce the number of long range links to a constant number, and still have relatively efficient routing: $O(\log^2 N)$ hops [19] or even $O(\log N)$ hops [18].

However, let us remember that we are interested in the flows and not the overlay. In fact, we can modify our model to have a *single* long range flow per node, and still have $O(\log^2 N)$ hops for routing messages in expectation. The trick is to create clusters of $\log N$ consecutive nodes in the ring, and allow each node to actually use only one of its long range links. Routing proceeds as usual: but when a node decides to send a message over a long range link, it first has to locate the node in its cluster that is allowed to use that link and send the message to that node along the ring. Note that nodes that are in the same cluster can rely on an identical set of long range links since clusters can be interpreted as replicas of a node in an overlay of size $N/\log N$.

Finally, we state without proof that a much simpler stochastic approach in which we have no clustering at all, but where each node can use only one random long range link results in a similar routing complexity in expectation. Here a node has to look at its $\log N$ neighbors in the ring and pick the best long range link that is allowed in some of these neighbors.

In sum, from the point of view of flows all nodes now have two ring flows (one in and one out) and two long range flows on average (one out and one in on average).

4.2.2 Locality

We can also optimize the ring by trying to assign IDs to nodes in such a way that the resulting ring has links which touch the smallest possible number of routers. Several algorithms are known for achieving such optimized topologies that could be adapted to this application, for example [9, 20].

5 Simulation Experiments

To examine the partial TDGs as seen locally from several points of the Internet, we (i) created a static AS-level model of the Internet topology and routing, (ii) mapped the overlay network to this AS-level topology, (iii) and we analyzed the local TDGs that are defined for each AS by the set of traversing flows in our model. We will now elaborate on these steps.

5.1 The AS-level Underlay

As our AS-level underlay we used an AS link dataset from CAIDA [6] that we cleaned by deleting uncertain links (around 3% of all links). We are aware of the methodological problems with collecting AS-level links and simulating protocols over them. However, for the purposes of this study, the main goal was not to achieve perfect low level realism but to capture the important structural properties of the Internet as a complex network, a level that even a good topology generator could provide.

We calculated the shortest paths for each pair of nodes in the topology after assuming that edges have equal weights. As a simple model of BGP routing we assumed that flows actually follow these shortest paths. Shortest paths also define the betweenness centrality of each node, that is, the number of shortest paths that touch a given node. This is a very important metric from the point of view of TDGs, since an AS with a high betweenness value is likely to be able to capture a more complete view of the TDG of the application.

The statistical properties of the AS graph have been studied intensively (see for example [17]). Figure 1 illustrates the distribution of betweenness centrality in the dataset we used, which is presented because we found the sharp switch to an exponential relationship at rank 1700 interesting.

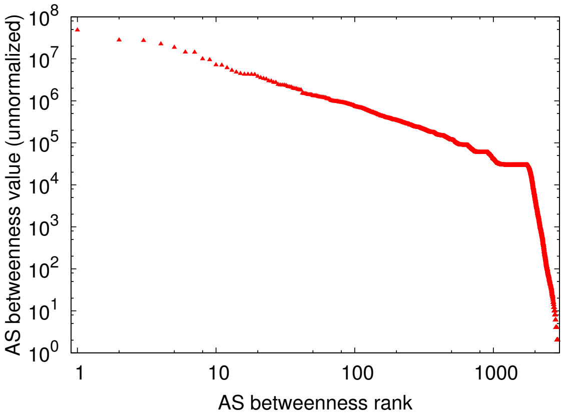


Figure 1: The power law relationship of betweenness rank and value in our dataset. After the sharp drop at around rank 1700, the relationship becomes exponential (not shown).

5.2 Mapping the Overlay to the Underlay

Based on notions described in Section 4, we experiment with two kinds of mappings: random and localized. First we describe the common settings for these two mappings, and then we discuss the specifics of both.

5.2.1 Common Settings

In all our experiments the overlay contains 100,000 nodes. Note that we do not expect our results to be sensitive to increasing the overlay size, since the overlay localization techniques we discussed in Section 4 essentially cause the problem to depend only on the AS-level graph.

The AS topology contains 14,630 nodes. With each type of overlay we map the overlay nodes onto the AS nodes in such a way that the number of overlay nodes in each AS is proportional to the size of the AS, but each AS has at least one overlay node. That is, in our model we do not take into account the geographical, social, or cultural bias that is known to affect botnet distribution [5]. The size of an AS is approximated based on the IP-prefix-to-AS mapping available from CAIDA[1].

It is interesting to note that the size and the betweenness of an AS seem to show no correlation, as illustrated by the scatter plot in Figure 2.

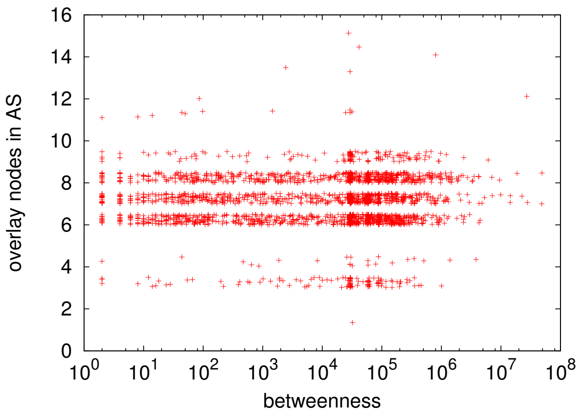


Figure 2: Scatter plot of the number of overlay nodes in an AS and the betweenness centrality of the same AS.

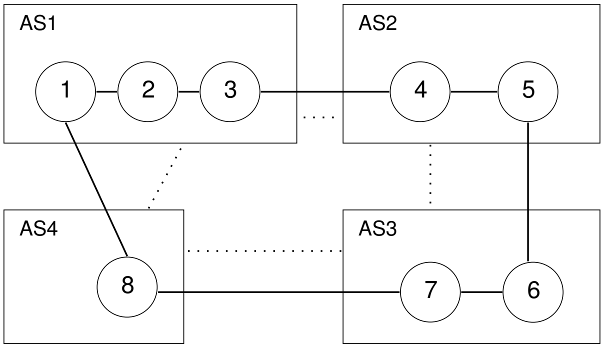


Figure 3: Localized mapping: alignment of the overlay ring (solid lines and circles) with the AS tour (dotted lines and rectangles) that is output by the nearest neighbor heuristic.

5.2.2 Specific Mappings

In the *random mapping* we assign overlay nodes to ASes at random, keeping only size-proportionality in mind, as outlined above.

To create a *localized mapping* as described in Section 4.2.2, we first define a traveling salesperson problem (TSP) over the AS topology and, using a simple heuristic algorithm, we produce a "good enough" tour over the ASes. Finally, we assign the overlay nodes to ASes in such a way that the overlay ring is consistent with this tour as illustrated in Figure 3.

We define the TSP as follows: find a permutation of the ASes such that if we visit all the ASes exactly once in the order given by the permutation, but assuming a closed tour that returns to the origin, and assuming also that for each transition from one AS to the other we follow the shortest paths in the AS-level topology, then the *sum of the hops* in the AS-level topology is minimal.

The heuristic we applied is nearest neighbor tour construction [11]. We start the tour with a random AS, and iteratively extend the tour by adding an AS that has the smallest shortest path length among those ASes that have not yet been visited. Ties are broken at random.

Before moving on to the analysis of TDGs, some comments are in order. First, our simulation is completely indifferent to the way a solution for the TSP problem is generated (i.e., a P2P algorithm or some other arbitrary heuristic method). What we would like to focus on is what happens when the mapping is sufficiently well localized.

Second, the heuristic mapping we produce is most likely quite far away from the optimal localization. The actual optimal mapping is prohibitively expensive to calculate since the TSP problem is NP-hard in general, and we have a very large instance. Moreover, the definition of the localization problem itself could be refined as well, taking the requirements of the P2P botnet into account more directly in the objective function, and, for example, minimizing the sum or the maximal number of flows that can be seen at the ASes.

For these reasons our results should be interpreted as an upper bound on the amount of information that is available at local nodes.

5.3 Analysis of TDGs

We experimented with four overlay models that are given by the two kinds of mappings described in Section 5.2.2 (random and localized) with or without the clustering technique described in Section 4.2.1.

For these models we simply collected the flows that traverse a given AS, created the TDG, and collected statistics. The statistics we collected were the following: number of nodes, number of edges, number of weakly connected components, size of largest weakly connected component, average node degree (where we count both incoming and outgoing connections) and finally, a metric called InO, introduced in [8]. InO is the proportion of nodes that have both incoming and outgoing connections.

The results are shown in Figure 4. The first observation we can make is that the more efficient factor for hiding the overlay traffic is clustering. Recall, that the main effect of clustering is to reduce the flows each node participates in from $O(\log N)$ to 4 on average. The effect of localization is significant as well, but it is less dramatic overall. There is one exception: the largest connected component, where localization results in a value that is two orders of magnitude smaller for the two most central ASes.

Let us first compare these results to those found in [8] for existing P2P networks in real traces. There, it was concluded that P2P traffic can be characterized by a high InO value (larger than 1%) and a high average degree

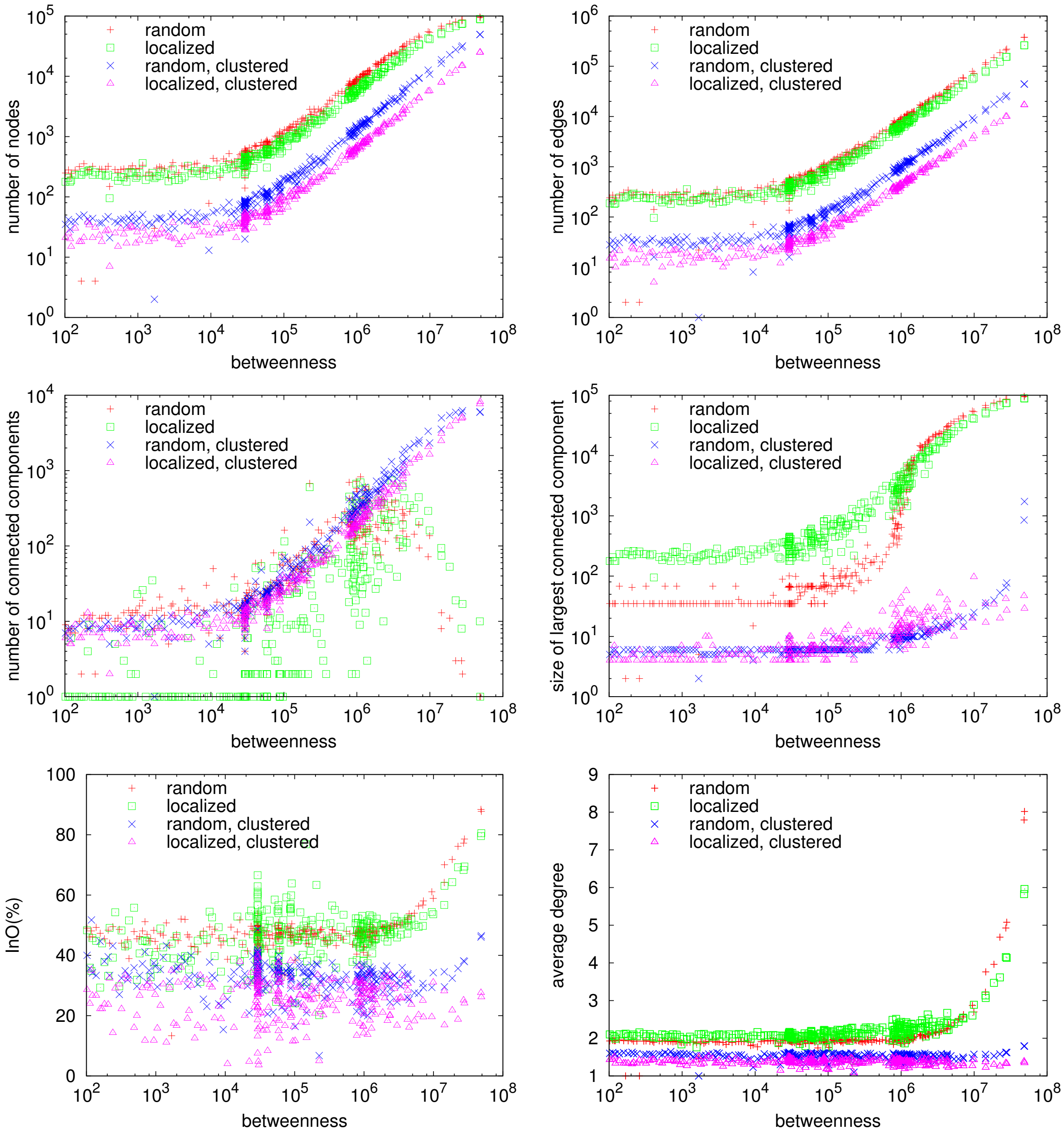


Figure 4: Different characteristics of the TDG as seen from different ASes of a given betweenness.

(larger than 2.8). From this point of view, the TDGs we observe do *not classify as P2P traffic*, because the average degree is extremely low: in fact less than 2 in the case of the localized and clustered network even for the most central ASes. It is interesting that even for the random mapping without clustering the threshold is crossed only at the most central ASes, although by a large margin.

On the other hand, the InO values are high. This is simply because we did not pay any attention to deceiving this metric explicitly. The reason is that in practice determining the direction of a flow is not very reliable, is prone to errors and quite possible to manipulate. We predict that the InO value could also be manipulated by a botnet using techniques that cannot be captured by the relatively high level model we apply that ignores flow details and dynamics.

In addition, in [8] some applications with high InO and low average degree have been found: one example is FTP, where the server initiates connections to the client as well, which further complicates detection and offers the botnet other opportunities for camouflage.

Of course it is possible that other metrics could help characterize these TDGs as belonging to a P2P network. Let us look at the TDGs using other metrics in order to have a more precise idea of what information is visible locally. Out of the 200,000 edges in the overlay, even the most central AS can see only 16,814 edges. The number of nodes in the TDG is 24,985 which is much larger than the number of edges: indeed, the connected components are mostly of size 2 (pairs) and 3. There are 8,172 clusters, the maximal of which contains only 29 nodes. A visualization of the TDG that belongs to the most central AS is shown in Figure 5. The information available at the less central ASes is significantly less, as shown in Figures 4 and 5. Finally, the maximal node degree we observed in any TDG we have generated is no more than 4.

It is important to emphasize that results presented here are based on the assumption that within one AS transit traffic traces can be aggregated and treated in a unified way. Although not impossible, this is a rather strong assumption, especially for the most interesting ASes with high betweenness centrality, that handle enormous volumes of transit traffic. In practice, the information visible locally could be even more fragmented.

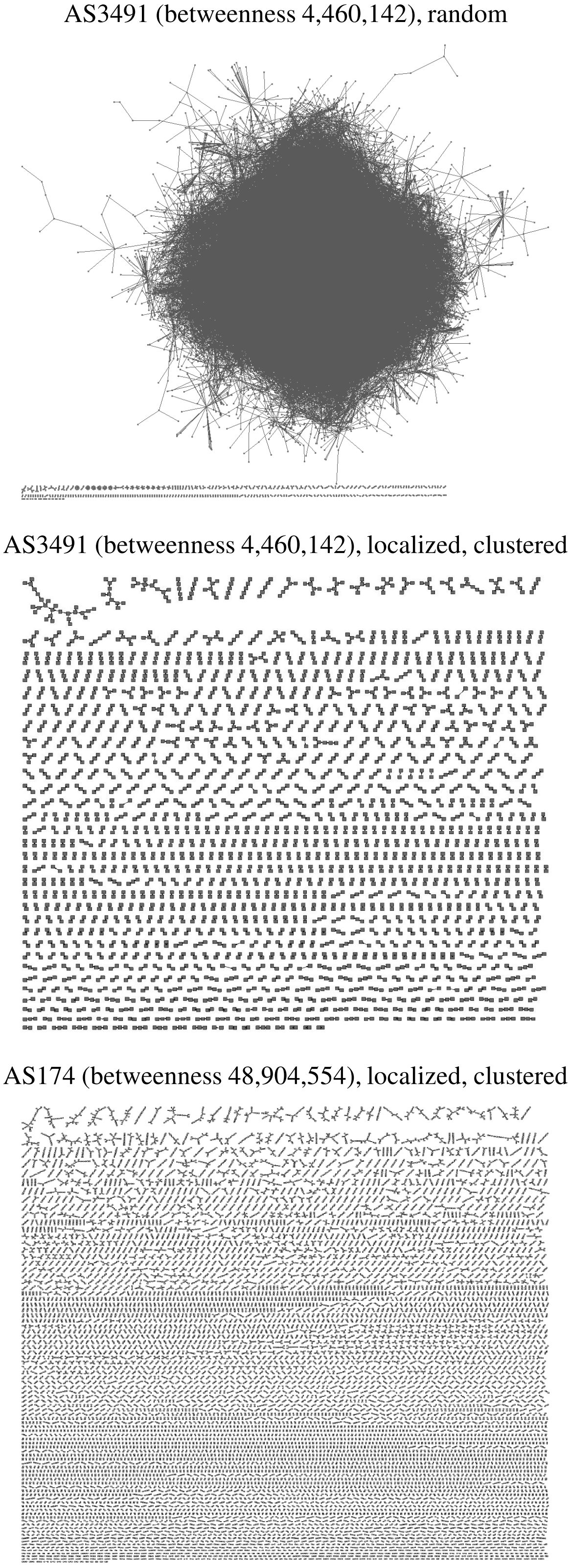
Overall, then, we may conclude that when localization and clustering are applied, the overlay network traffic is almost completely hidden. A non-trivial proportion of the traffic can be seen only at the most central ASes, but even there, what is visible is predominantly unstructured.

6 Conclusions

Instead of looking at existing P2P botnets, we created synthetic flow data to model the set of flows that are available at an AS locally for observation. While it is clear that this methodology involves a simplified model of communication, it does allow us to get ahead of botnets via experimenting with algorithms that have not yet been deployed.

In spite of the low resolution that this methodology offers, we were able to predict and analyze a real problem: P2P overlays that are capable of efficiently and robustly organizing and controlling a large set of bots with a minimal communication footprint so as to avoid automated detection.

We hope that our results will provide non-trivial clues concerning directions for future research in automated botnet detection. Our results also show that we need to fight fire with fire and develop and apply P2P technology over large sets of cooperating administrative domains.



AS3491 (betweenness 4,460,142), random

AS3491 (betweenness 4,460,142), localized, clustered

AS174 (betweenness 48,904,554), localized, clustered

Figure 5: Visualizations of TDGs at various ASes. AS174 had maximal betweenness in the dataset.

7 Acknowledgments

M. Jelasity was supported by the Bolyai Scholarship of the Hungarian Academy of Sciences.

References

[1] Cheetancheri, S. G., Agosta, J. M., Dash, D. H., Levitt, K. N., Rowe, J., and Schooler, E. M. A distributed host-based worm detection system. In *Proceedings of the 2006 SIGCOMM workshop on Large-scale attack defense (LSAD'06)* (New York, NY, USA, 2006), ACM, pp. 107–113.

[2] Dagon, D. Botnet detection and response: The network is the infection, 2005. OARC Workshop presentation.

[3] Grizzard, J., Sharma, V., Nunnery, C., Kang, B., and Dagon, D. Peer-to-peer botnets: Overview and case study. In *Proceedings of the First USENIX Workshop on Hot Topics in Understanding Botnets (HotBots'07)* (2007).

[4] Gu, G., Perdisci, R., Zhang, J., and Lee, W. BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proceedings of the 17th USENIX Security Symposium (Security'08)* (2008).

[5] Holz, T., Steiner, M., Dahl, F., Biersack, E., and Freiling, F. Measurements and mitigation of peer-to-peer-based botnets: a case study on storm worm. In *Proceedings of the 1st USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'08)* (Berkeley, CA, USA, 2008), USENIX Association.

[6] Hyun, Y., Huffaker, B., Andersen, D., Aben, E., Luckie, M., k claffy, and Shannon, C. The IPv4 Routed /24 AS Links Dataset – 2008-01-02. `http://www.caida.org/data/active/ipv4_routed_topology_aslinks_dataset.xml`.

[7] Iliofotou, M., Pappu, P., Faloutsos, M., Mitzenmacher, M., Singh, S., and Varghese, G. Network monitoring using traffic dispersion graphs (TDGs). In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement (IMC'07)* (New York, NY, USA, 2007), ACM, pp. 315–320.

[8] Iliofotou, M., Pappu, P., Faloutsos, M., Mitzenmacher, M., Varghese, G., and Kim, H. Graption: Automated detection of P2P applications using traffic dispersion graphs (TDGs). Tech. Rep. UCR-CS-2008-06080, Department of Computer Science and Engineering, University of California, Riverside, June 2008.

[9] Jelasity, M., and Babaoglu, O. T-Man: Gossip-based overlay topology management. In *Engineering Self-Organising Systems: Third International Workshop (ESOA 2005), Revised Selected Papers* (2006), S. A. Brueckner, G. Di Marzo Serugendo, D. Hales, and F. Zambonelli, Eds., vol. 3910 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 1–15.

[10] Jelasity, M., Voulgaris, S., Guerraoui, R., Kermarrec, A.-M., and van Steen, M. Gossip-based peer sampling. *ACM Transactions on Computer Systems 25*, 3 (Aug. 2007), 8.

[11] Johnson, D. S., and McGeoch, L. A. The traveling salesman problem: A case study in local optimization. In *Local Search in Combinatorial Optimization*, E. H. L. Aarts and J. K. Lenstra, Eds. John Wiley and Sons, 1997, pp. 215–310.

[12] Kanich, C., Levchenko, K., Enright, B., Voelker, G. M., and Savage, S. The heisenbot uncertainty problem: Challenges in separating bots from chaff. In *Proceedings of the 1st USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'08)* (Berkeley, CA, USA, 2008), USENIX Association.

[13] Kermarrec, A.-M., and van Steen, M., Eds. *ACM SIGOPS Operating Systems Review 41*. Oct. 2007. Special issue on Gossip-Based Networking.

[14] Kleinberg, J. Navigation in a small world. *Nature 406* (2000), 845.

[15] Kleinberg, J. The wireless epidemic. *Nature 449* (2007), 287–288. News and Views.

[16] Lua, E. K., Crowcroft, J., Pias, M., Sharma, R., and Lim, S. A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys and Tutorials 7*, 2 (2005), 72–93.

[17] Mahadevan, P., Krioukov, D., Fomenkov, M., Dimitropoulos, X., k c claffy, and Vahdat, A. The Internet AS-level topology: three data sources and one definitive metric. *SIGCOMM Comput. Commun. Rev. 36*, 1 (2006), 17–26.

[18] Malkhi, D., Naor, M., and Ratajczak, D. Viceroy: A scalable and dynamic emulation of the butterfly. In *Proceedings of the 21st ACM Symposium on Principles of Distributed Computing (PODC'02)* (2002).

[19] Manku, G. S., Bawa, M., and Raghavan, P. Symphony: Distributed hashing in a small world. In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems (USITS'03)* (2003).

[20] Massoulié, L., Kermarrec, A.-M., and Ganesh, A. J. Network awareness and failure resilience in self-organising overlay networks. In *Proceedings of the 22nd Symposium on Reliable Distributed Systems (SRDS 2003)* (Florence, Italy, 2003), pp. 47–55.

[21] Montresor, A. A robust protocol for building superpeer overlay topologies. In *Proceedings of the 4th IEEE International Conference on Peer-to-Peer Computing (P2P'04)* (Zurich, Switzerland, Aug. 2004), IEEE Computer Society, pp. 202–209.

[22] Nguyen, T. T. T., and Armitage, G. A survey of techniques for Internet traffic classification using machine learning. *IEEE Communications Surveys and Tutorials 10*, 4 (2008), 56–76.

[23] Ramachandran, A., Feamster, N., and Dagon, D. Revealing botnet membership using DNSBL counter-intelligence. In *Proceedings of the 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'06)* (2006).

[24] Ripeanu, M., Iamnitchi, A., and Foster, I. Mapping the gnutella network. *IEEE Internet Computing 6*, 1 (2002), 50–57.

[25] Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., and Balakrishnan, H. Chord: A scalable peer-to-peer lookup service for Internet applications. In *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)* (San Diego, CA, 2001), ACM, ACM Press, pp. 149–160.

[26] Strayer, W. T., Lapsely, D., Walsh, R., and Livadas, C. Botnet detection based on network behavior. In *Botnet Detection: Countering the Largest Security Threat*, W. Lee, C. Wang, and D. Dagon, Eds., vol. 36 of *Advances in Information Security*. Springer, 2008, pp. 1–24.

[27] Weaver, N., Ellis, D., Staniford, S., and Paxson, V. Worms vs. perimeters: the case for hard-LANs. In *Proceedings of the 12th Annual IEEE Symposium on High Performance Interconnects (HOTI'04)* (Washington, DC, USA, 2004), IEEE Computer Society, pp. 70–76.

Notes

[1] `http://www.caida.org/data/routing/routeviews-prefix2as.xml`