# Intrusion as (Anti)social Communication: Characterization and Detection

Qi Ding
Boston University
qiding@math.bu.edu

Natallia Katenka
Boston University
nkatenka@math.bu.edu

Paul Barford
University of Wisconsin
pb@cs.wisc.edu

Eric Kolaczyk
Boston University
kolaczyk@math.bu.edu

Mark Crovella
Boston Univeristy
crovella@cs.bu.edu

## ABSTRACT

A reasonable definition of intrusion is: entering a community to which one does not belong. This suggests that in a network, intrusion attempts may be detected by looking for communication that does not respect community boundaries. In this paper, we examine the utility of this concept for identifying malicious network sources. In particular, our goal is to explore whether this concept allows a core-network operator using flow data to augment signature-based systems located at network edges. We show that simple measures of communities can be defined for flow data that allow a remarkably effective level of intrusion detection simply by looking for flows that do not respect those communities. We validate our approach using labeled intrusion attempt data collected at a large number of edge networks. Our results suggest that community-based methods can offer an important additional dimension for intrusion detection systems.

## Categories and Subject Descriptors:

C.2.3 [**Network Operations**]: Network Monitoring, C.4 [**Performance of Systems**]: Modeling Techniques, E.1 [**Data**]: Data Structures

## General Terms:

Algorithms, Design, Experimentation, Measurement, Security

## Keywords:

Intrusion detection, Social graphs

## 1. INTRODUCTION

Network-based intrusion detection and prevention systems (IDS/IPS) play a significant role in protecting IT assets from malicious attacks. Nonetheless, it has been known for some time that attackers can often easily defeat IDS/IPS systems [19]. The difficulty of on-going identification of new vulnerabilities coupled with the explosion in diversity of malware makes it virtually impossible to keep standard signature-based intrusion detection and prevention systems [16, 18] configured with rule sets that protect against previously-unseen attacks. This inherent challenge speaks directly to the need for complementary methods for network-based attack detection.

Anomaly-based network attack detection offers a compelling alternative to signature-based methods. The conceptual framework for network anomaly detection depends on defining characteristics of traffic that are considered "normal," and labeling as anomalous the traffic that deviates significantly from normality. In this way anomaly detectors have the potential to identify unanticipated or previously unseen attacks that would otherwise be invisible to signature-based detectors.

Despite their appeal, there are significant barriers to the practical use of anomaly-based detectors in IT security infrastructures. In particular, anomaly detection relies on formulating a robust definition of normality that sharply separates wanted from unwanted traffic. This has proven to be difficult in the past, in particular for detection systems that consider only simple features such as traffic volume [17]. Thus, we argue that new, creative methods for defining the "essence" of normal behavior are required in order to begin to realize the promise of anomaly detection. In this paper we examine the utility of one such definition for the specific problem of intrusion detection.

We start from the hypothesis that intrusion is *entering a community to which one does not belong.* The advantage of this starting point for intrusion detection is that it is rooted in a community-centric essence of normality. That is, rather than defining intrusion in terms of locations in physical space (where normality is harder to characterize), this hypothesis defines intrusion in terms of locations in social space, where normality often has well-understood properties. In fact, different network applications have clearly-identifiable communication patterns (*e.g.,* P2P systems), and these patterns can reflect the social behavior of users [8].

To connect our hypothesis to network traffic, we consider whether communication patterns between end hosts (identified by IP address) as evidenced in network flow data can form the basis for community detection, and whether the resulting communities form an effective basis for intrusion detection. A standard representation of communication patterns in network traffic is the bipartite graph, in which traffic sources and traffic destinations form the two vertex sets, and directed edges denote traffic flow from sources to destinations. In addition to this standard view, we explore the utility of a simpler graph representation in which vertices correspond only to traffic sources. This simpler representation admits a number of variants that differ in their level of detail.

Armed with this range of representational choices, we examine the characteristics of graph nodes using `netflow` data collected from a large European ISP. We use firewall and IDS/IPS logs pro-


*KDD'12,* August 12–16, 2012, Beijing, China.

vided by Dshield.org [22] to identify malicious nodes in our data corpus. We show that malicious nodes do indeed exhibit distinctive properties across a range of graph representations. We begin by considering the use of cardinality [9] as a tool to identify malicious nodes. We show that, while cardinality can identify some malicious nodes, it misses the majority of malicious nodes. Furthermore, it is clear that a detector based on high cardinality is easily overcome by attacks that keep the number of outgoing connections below a normal-seeming threshold.

Therefore, we go beyond the simple cardinality measure to examine features that are more specific to communication that disregards community boundaries – *i.e.,* metrics of antisocial behavior. This definition is critical in the security context because it significantly increases the difficulty of obviation by an attacker whose connection behavior is inconsistent with the community. We propose a simple proxy for a node that seems to disregard communities in our graphs: the *cut-vertex.* We show that, in our data, cut-vertices are very likely to be malicious. Unfortunately, the condition of being a cut-vertex is a fragile one, and depends strongly on the density of communication and the degree of sampling present in the underlying data. As the volume of data analyzed increases, cut-vertices become hidden.

This motivates us to generalize and make robust the concept of a cut-vertex using concepts from the social networking literature. We hypothesize that natural extensions of the cut-vertex concept can be based on local clustering coefficient and betweenness centrality. In fact, we show that low local clustering coeffecient and high betweenness centrality are strongly associated with malicious sources in certain graph representations. We explore these metrics by considering their application to our set of different network representations, from simple to detailed.

Our results evaluate the relative utility of different graph representations, and of different metrics on those graphs, for capturing antisocial behavior as it relates to network intrusion. We show that clustering is a particularly useful metric in this regard, while the proper choice of graph representation (simple versus detailed) is less crucial. Our results show that the proper choice of graph and metric allows us to detect a large fraction of malicious sources, even among those sources that are low-degree and therefore undetectable using traditional methods. Taken in total, these results are strongly suggestive that community-based analysis can shed significant light on the nature of malicious traffic sources.

## 2. DATA

In our study we use `netflow` data collected from a large European ISP. We focus on traffic collected at one router over a range of days from May 1, 2007 to May 14, 2007. For our examples, results are often for a single day (May 1) only; for the full evaluation of our methods, results are given for the period spanning all 14 days.

An important aspect of our work is that all of our `netflow` data is based on 1/1000 packet sampling. This level of packet sampling is typical of a large ISP, and we consider it a requirement of practical intrusion detection methods to be effective despite the information loss dictated by low sampling rates. For example, a single day's traffic (May 1) consists of 29,585 sources sending 2,033,520 flows to 185,145 destinations.

In any study of proposed intrusion detection methods, validation is both crucial and problematic. Traffic containing known, labeled intrusion attempts is rare or else severely outdated [12]. Hence we take a novel approach to identifying intruders in our data. We obtain from the SANS Institute the DShield logs for each day covered by our data [22]. DShield collects firewall and intrusion detection system logs from over 1700 networks world wide. The logs provide a summary of malicious and unwanted activity in each of the provider's networks. Each log entry includes, among other information, the unobfuscated source IP address of the packet that triggered the entry. DShield logs have been used in a number of prior studies of global malicious activity [23, 2].

For each day's traffic, we identify those traffic sources that also attempted an intrusion as recorded by DShield (we call these *DShield sources*). While not all of the traffic from DShield sources is sure to be intrusion attempts, we show in Section 4 that the vast majority shows characteristics suggestive of intrusion. Thus we make the approximation that if a source appears in DShield, its activity in our traffic data on the same day is malicious. While this is far from a perfect form of ground truth, we believe that inaccuracies are not likely to strongly influence our final results. This is because our primary results concern the comparison of detection methods (rather than absolute detection rates).

For example, in our May 1 data, 329 sources recorded in DShield logs also appear in our traffic. Note however that these 329 sources sent out 392,723 distinct flows to 66,618 distinct destinations. This sort of activity is highly suggestive of intrusion attempts. We provide more detailed evidence of the malicious nature of DShield sources in Section 4.

## 3. NETWORK REPRESENTATIONS

To begin, we note that our motivating hypothesis – *i.e.,* that intrusion is entering a community to which one does not belong – requires us to identify when one 'belongs' (and when one does not 'belong') to a 'community.' Both of these terms are relational in nature, and so it is natural to employ network-based representations of our data for our work. Generally speaking, any choice of network representation needs to be made carefully and can be expected to influence the level of success one achieves with high-level tasks utilizing such networks (*e.g.,* [13, Ch 3]). For that reason, one of the goals of our study is to examine certain alternatives in choice of network representation and understand the implications of this choice on our ability to detect intrusions.

We begin by defining a standard bipartite graph representation of communication patterns in computer network traffic. Specifically, let $G_B = (V, U, E_B)$, where vertices $v \in V$ represent source IPs (srcIP), vertices $u \in U$ represent destination IPs (dstIP), and edges $\{v, u\} \in E_B$ represent data flows from sources $v$ to destinations $u$. In constructing such a graph $G_B$ from data, we assign a vertex $v \in V$ for every unique IP address that played a role as a source, and a vertex $u \in U$ for every unique IP address that played a role as a destination. We assign an edge $\{v, u\}$ to $E_B$ if and only if there is a flow in our data from $v$ to $u$. In principle, multiple edges arise when there are several flows between address pairs. However, we find it convenient in what follows to simplify this structure by assigning edge weights equal to the multiplicity of an edge.

Figure 1 shows an illustration of such a graph $G_B$, consisting of a small connected component extracted from our data. There are 10 source nodes (*i.e.,* nodes 1 through 10), 4 destination nodes (*i.e.,* nodes $a$ through $d$), and 13 edges, corresponding to 25 flows, with weights ranging from 1 to 4. Note that sources share destinations to varying extents. For example, sources $1, 2, 3$, and 10 all talk only to destination $a$, while sources 4 and 6 talk only to destination $c$. This pattern suggests some common purpose to the traffic in each of the two subsets of corresponding flows, and hence suggests in turn that some notion of 'community' may apply to these two sets of sources. In contrast, while source 9 also talks to destination $a$ and source 7 also talks to destination $c$, they each talk as well to destination $b$. This observation suggests that, while sources 7 and 9 participate in the communities defined around destinations $a$ and

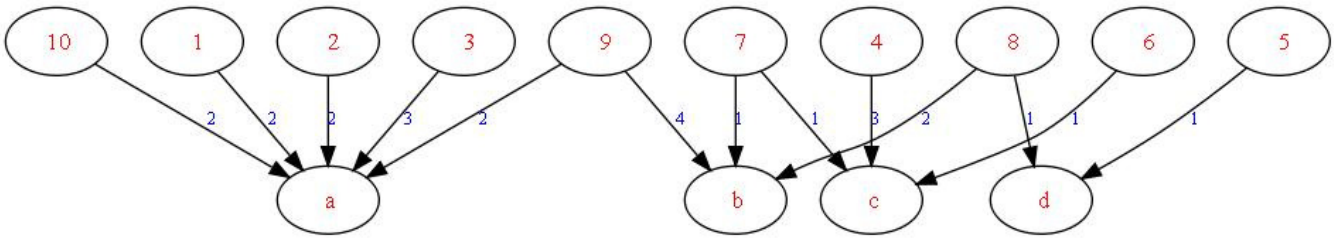


**Figure 1: Bipartite graph representation of network flow traffic.**

$c$, they do not necessarily belong to those communities. Hence, we say that their role is both 'social' and 'anti-social' in nature. It is this behavior that we hypothesize may usefully be associated with intrusion.

While $G_B$ is a standard representation of network traffic, it is difficult to analyze for antisocial behavior because nodes in $G_B$ are not all of the same type: some are traffic sources, and some are traffic destinations. Since our focus is on identifying sources of malicious traffic, a natural way to simplify the problem is to consider graphs in which vertices correspond only to sources. These networks are essentially reductions of the bipartite graph $G_B$. These reductions may be defined in a number of alternative ways, and the alternatives can vary in detail. We consider two such alternatives in this study.

The first is a standard one-mode projection of $G_B$, in the form of an undirected graph $G_P = (V, E_P)$, where nodes $v_i$ and $v_j$ are connected if and only if they share at least one common destination. The one-mode projection of the bipartite graph in Figure 1 is shown in Figure 2. Note that under this representation the types of 'communities' we identified, *i.e.,* source nodes that all communicate with a common destination, exhibit a distinct topological structure in $G_P$, in the form of cliques. For example, nodes $1, 2, 3, 9$, and 10 form a five-clique, while nodes $4, 6$, and 7 form a three-clique. Note too that nodes 7 and 9, which we identified as exhibiting an example of antisocial behavior, stand out clearly as being both members of their respective cliques and, at the same time, connected by a single edge to each other's cliques.

This edge is an example of a *bridge*, in that its removal increases the number of components in the graph. Similarly, these two vertices are examples of a *cut-vertex*. These two complementary notions are popular in the social network literature (*e.g.,* [6, Sec 7.3]), where they are used, among other things in the study of diffusion of information. Here we maintain that, if cliques in our network $G_P$ capture the traffic common to Internet communities sufficiently well, and if bridges and cut-vertices capture antisocial behavior on the fringe of these communities, then this illustration suggests that identification of communities and the bridges/cut-vertices between them may be used to detect source IPs sending intrusive traffic.

A potential drawback of the one-mode projection $G_P$ as a source-based view of the communication underlying our network flow data is the loss of certain information. For example, regardless of whether two source nodes send traffic to a single common destination or 100 common destinations, only a single edge will be placed between them in $G_P$. Furthermore, it can be argued that the notion of communication in a community being determined solely by sending traffic to a common destination is rather limited, and that real communication is decidedly more nuanced. For example, if sources $v_1$ and $v_2$ both send traffic to destination $a$, and $v_2$ and another source $v_3$ both send traffic to a different destination $b$, it may be that all of $v_1, v_2$ and $v_3$ belong to the same community. In other words, what is important in determining source-based communities may be a sense of how similar a set of sources are to each other in the bipartite communication graph $G_B$.

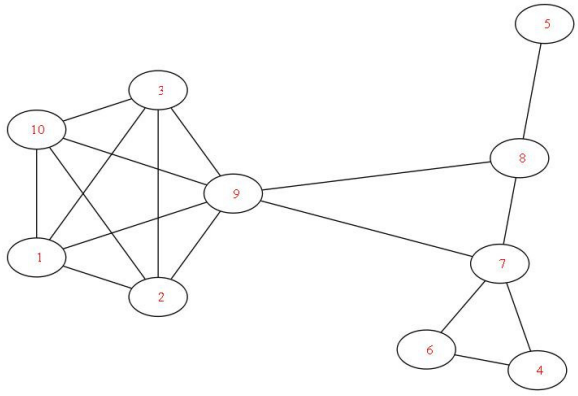


**Figure 2: One-mode projection.**

## 4. CHARACTERIZING MALICIOUS SOURCES

A number of previous studies have examined how the properties of communication graphs can be used to identify malicious sources. The main approach taken to date has been to simply observe that malicious sources often send traffic to many other nodes, and so candidates for detection are 'high cardinality' nodes – in the language of the previous section, those with high out-degree in $G_B$ [9, 11, 3, 8].

Unfortunately, attackers now routinely seek to scan or probe without raising cardinality alerts, and detection based on source out-degree has to be tuned quite carefully [10]. We can observe this effect in our data. Figure 3 shows the histogram of the base-10 log of out-degree for all of the Dshield sources in our May 1 dataset. The figure shows that malicious sources have out-degrees spanning a vast range; while some sources send to thousands of destinations, many send to only hundreds or tens of destinations.

To illustrate the problem this causes for degree-based detectors, we evaluated the performance of a degree-based detector on our data. Such a detector is configured by choosing the false alarm rate that is considered tolerable by the operator. We chose false alarm rates of 5% and 2.5%; these are in practice quite high, but doing so presents the detection ability of the degree-based method in the best possible light.

Nonetheless, even when high false alarm rates are tolerated, the degree-based detector fails to detect a significant fraction of the

malicous sources. The thresholds for degree-based detection at false alarm rates of 5% and 2.5% are shown in Figure 3. For each threshold, sources to the right of the line are those that would be detected. The figure shows that much more than half (64% and 79%) of the malicious sources go undetected using the degree-based method.

This observation motivates our interest in developing detection methods that complement pure cardinality-based approaches. That is, we seek methods that can identify potential attackers despite the fact that the attacker does not send a large number of probes in the observation period (and so does not have high out-degree).

To that end, we turn to the notion of antisocial communication as a potentially more sophisticated indicator of network intrusion. To do so, we need to define metrics that can identify antisocial nodes within the graphs $G_P$ and $G_W$ defined in the previous section. In the remainder of this section we present such metrics. Then, in the next section, we will show simple detection algorithms based on these metrics that are remarkably accurate in identifying malicious sources among those sources that are *not* high-cardinality hosts.

To illustrate the metrics we define, in this section we use flow data from just the first of our 14 days (*i.e.,* May 1st, 2007). In addition, for reasons of computational efficiency, we use a subset of the network flow data for that day, based on the set of all flows sent from a simple random sample of size 10K selected from those source IP addresses active that day in our dataset. The flows obtained in this manner induced a corresponding sampling of destination IP addresses. Completing our sampled source and destination lists with those destinations and sources obtained in the original stage that also acted as sources and destinations, respectively, we are left with a subset of the data whose bipartite graph representation $G_B = (V, U, E_B)$ has $|V| = 10,897$, $|U| = 53,877$, and $|E_B| = 879,010$. From this we then created the binary and weighted one-mode projections of $G_B$, *i.e.,* $G_P = (V, E_P)$ and $G_W = (V, E_W)$, respectively; where $|V| = 10,897$ (being the same set $V$ defining $G_B$) and $|E_P| = |E_W| = 766,680$. The results below for these graphs are representative of what we observed over multiple simulation trials, over multiple days, as we discuss later, in Section 5.

Importantly, for our purposes, among the $10,897$ sources in our networks $G_B$, $G_P$, and $G_W$, 182 were identified as malicious by examination of the DShield logs of that day. In what follows in this section, we focus on the challenge of characterizing these malicious source IPs given only the basic information on which IPs sent flows to which others. We explore three classes of techniques, based on ideas of community detection, cut-vertices, and local graph structure, respectively. The lessons we learn from this characterization study then lead naturally to the detection algorithms we present later in Section 5.

## 4.1 Community Detection

Our notion of (anti)social behavior in the source-based social networks defined in the previous section, and the reliance of this notion on 'communities' in those networks, suggests the relevance of so-called community detection methods from the social network literature in developing detection algorithms. There are many such methods of this sort. All of them essentially take as input a graph topology and they output a partitioning of the graph into subsets of nodes, for which nodes within subsets are more heavily linked than nodes between subsets. See [13, Ch 4.3.3] and references therein, for example.

A popular method in this area is that of Newman, Clauset and Moore [4], based on the concept of modularity. This method, as with others like it, attempts to optimize a certain complexity score (called the 'modularity') to produce as parsimonious a partition of the graph as possible. For illustration, we applied this algorithm to the largest connected component of the one-mode projection source graph $G_P$, consisting of 8480 source nodes (out of the $10,897$ sources in the full graph) and the edges among them. Among these 8480 nodes, 161 of those are DShield sources (the other 24 DShield sources are represented as singleton nodes in the full graph). The algorithm found an optimal clustering with a total of 68 clusters (or 'communities'), the size and number of which are summarized in Table 1.

**Table 1: Summary of clusters produced by standard community detection, as applied to the largest connected component of $G_P$.**

| Size of Cluster | # of Clusters | # of DShields |
|---|---|---|
| 6784 | 1 | 158 |
| 986 | 1 | 1 |
| 8 to 243 | 10 | 0 |
| $\leq 7$ | 56 | 2 |
| Total | 68 | 161 |

Unfortunately, the partitioning of our subgraph into these communities, relative to the distribution of the 161 malicious DShield source nodes, does little to reveal the latter in any obvious fashion, since 158 of these DShield source IPs are contained within the largest cluster. See Table 1. Hence, this particular method of defining 'communities' does little to distinguish the DShield sources from those communities upon which they intrude. In fact, this result is not entirely surprising, since it has been found that the maximum modularity partition suffers from a so-called 'resolution limit', in which it tends to fail to extract clusters smaller than a certain size, due to the particular manner in which the degree distribution of the underlying graph enters into the criterion function being optimized. See [7], for example.

## 4.2 Cut-Vertices

The problem with the standard community detection approach in this setting is that, in seeking a highly parsimonious partitioning, it is not operating at a sufficiently fine scale. While we could, of course, choose to explore partitions of the graph resulting from non-optimal choices of the objective function (*i.e.,* modularity), it is not clear how best to do so once optimality is left behind. Alternatively, therefore, we implemented a type of simple, bottom-up iterative pruning algorithm, which proved to be decidedly more powerful in isolating malicious nodes.

Specifically, and recalling the importance of cut-vertices in our illustrations of Section 3, as prototypical examples of how a node might participate in communication in (anti)social ways, we implemented a simple recursive seach-and-prune strategy for greedily finding such cut-vertices. Starting again with the largest connected component of the one-mode projection source graph $G_P$, we used a standard max-flow/min-cut algorithm to identify all cut-vertices in that component. These vertices were then removed, the act of which, by definition, broke the component into at least two separate connected components. We then took the larger of these connected components and repeated the process just described. This series of steps was iterated until we reached a largest connected component in which no further cut-vertices are found. This component had 4633 nodes, just over 50% the size of that in the first stage of the algorithm (i.e., 8480).

Note that at each stage of the algorithm, until convergence, the act of pruning induces additional cut-vertices, which in some sense

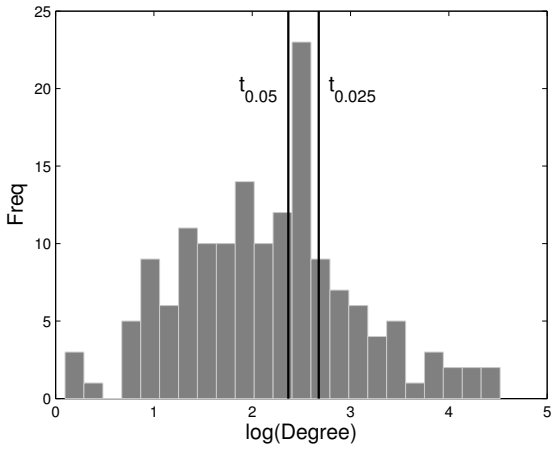


**Figure 3: Distribution of base-10 log of Dshield source out-degrees.**

therefore were 'nearly' cut-vertices at the previous iteration. So our algorithm can be viewed as a 'quick-and-dirty' way of discovering nodes where the notion of cut-vertex is applied in an increasingly robust manner. The output of our algorithm is a list of 1303 nodes. In examining this list, we find that 86 are DShield sources, *i.e.,* the proportion of Dshield sources among cut-vertices is about 6.6% . While this percentage may not seem large in itself, this is likely due in no small part to the greedy nature of our algorithm (*i.e.,* recall that at each iteration we work only on the largest connected component emerging from the component analyzed at the previous iteration). Importantly, however, a cross-validation study shows that this number is highly significant. Specifically, using the same network $G_P$, we randomly assigned the 161 labels of 'DShield' to arbitrary nodes in this graph and applied our algorithm. This process was repeated 100 times. The average proportion of pseudo-DShield nodes discovered over the 100 trials of this simulation was 1.66%, with a 95% confidence interval of $(1.06\%, 2.27\%)$. These results suggest that DShield source IPs are highly enriched among cut-vertices.

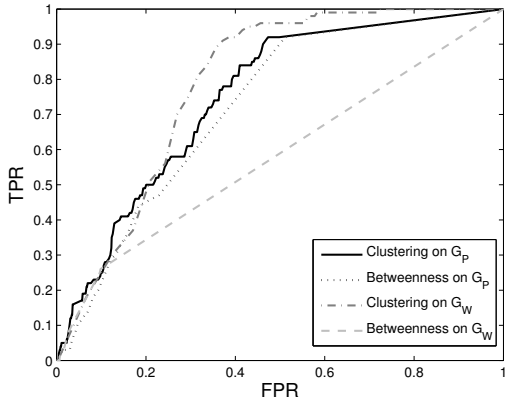


**Figure 4: Detection accuracy for Day 1.**

# 5. DETECTING MALICIOUS SOURCES

With an ability to distinguish between DShield and non-DShield nodes, we now demonstrate the use of clustering and betweenness on our source graphs for detection. We present first the results for the analysis of a single day of traffic, in detail, and then a summary of results for the same analysis over fourteen consecutive days of traffic.

The previous section showed that a sizeable fraction of malicious hosts can not be detected by simple degree-based methods. Hence, in all our analyses below, we focus only on detection of those nodes that can *not* be detected by degree-based methods. That is, we eliminate from our validation sets all those nodes having degree greater than the threshold shown in Figure 3 for the 5% false alarm rate.

It is important to recognize that high-degree malicious sources are easily detected using the antisocial metrics, because they communicate with so many unrelated communities. Hence, restricting our attention only to low-degree nodes makes our detection problem much harder, and focuses on a set of sources that are not detectable using standard (degree-based) methods.

## 5.1 Analysis of Day 1 Traffic

Our proposed intrusion detection system is simple and intuitive, in that it entails just the thresholding of the clustering and betweenness metrics introduced in the previous section in our characterization of malicious sources. To evaluate this approach, we do the following. Given a graph $G_P$ or $G_W$, constructed from a set of network flow data as described in Section 3, we calculate the clustering and betweenness values for all vertices in our validation set (*i.e.,* those not detected by a standard degree-based method, at 5% false alarm rate). These values are then compared to a threshold, across the appropriate range of thresholds, with a node being declared malicious if its clustering coefficient (betweenness centrality) is below (above) that threshold. Receiver operating characteristic (ROC) curves and the area under the curve (AUC) are used summarize performance.

To begin to understand the relative power of the two metrics (clustering and betweenness) and the two graph representations (the projection $G_P$ and the weighted projection $G_W$), we present detection results in Figure 4. This receiver operating characteristic (ROC) plot shows the tradeoff between false alarm rate (on the x

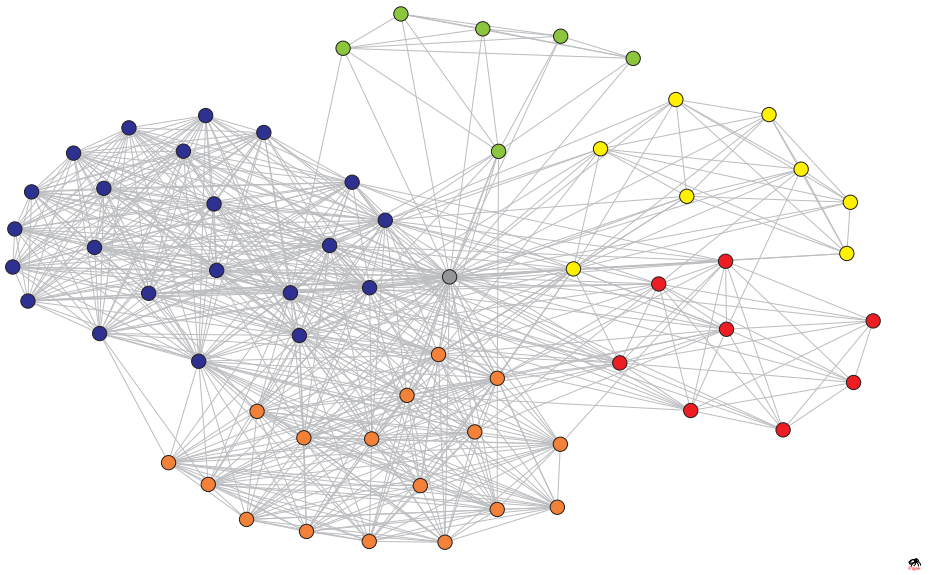

**Figure 5: Visualization of DShield neighborhood comprised of 62 nodes.**

axis) and detection rate (on the y axis) for the four combinations of metric and graph representation.

Overall, the figure shows that detection ability of the antisocial method for these difficult-to-detect sources is quite good. It also highlights distinctions among the approaches. The area under the curve (AUC) for these ROC curves, a single-number summary of accuracy, is higher for clustering than betweenness in both the binary graph $G_P$ (i.e., 0.75 versus 0.724) and the weighted graph $G_W$ (*i.e.,* 0.787 versus 0.581). These numbers – particularly the AUC of 0.787 for clustering on the weighted projection source graph $G_W$ – are quite encouraging, and point strongly to the utility of our use of social structures here.

To illustrate the nature of the antisocial communication detectable using these methods, we chose an arbitrary DShield source. We then extracted from the source graph $G_P$ the so-called 'egocentric' subgraph associated with this source, consisting of the source node, its 62 neighboring nodes, and those edges to and among these neighbors. Figure 5 shows a representation of the subgraph thus obtained. The Dshield node is depicted in the middle of what can be seen to be five natural clusters, indicated by different colors, identified using a standard community detection algorithm. The Dshield node was placed in its own cluster by the algorithm. The overall structure of the subgraph is consistent with our notion of (anti)social behaviour in the source-based social networks: the malicious source node is clearly 'entering' communities to which it does not 'belong.'

## 5.2 Analysis of 14-day Traffic

In order to verify the robustness of the 1-day results in Section 5.1 for our proposed intrusion detection system, we analyzed all fourteen days of the data described in Section 2. For each day, we repeated 10 times the process described in Section 4, wherein each time we generated a random bipartite graph $G_B$ from $10K$ randomly sampled source vertices and constructed the corresponding source projections $G_P$ and $G_W$. For each such pair of projection graphs, we calculated the local clustering and betweenness values

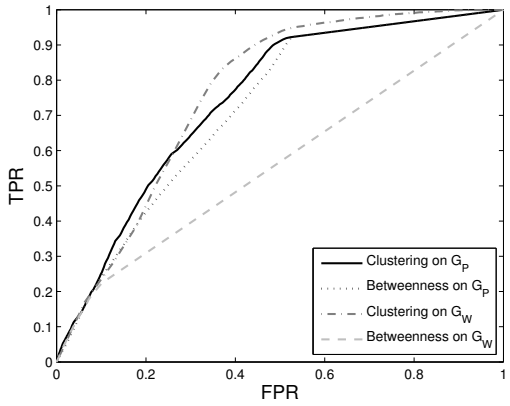


**Figure 6: Average ROC curves summarizing detection accuracy over 14 days.**

for all sources in components of size at least three (*i.e.,* we dropped all singletons and dyads from our analysis).

In Figure 6 we show the average of the resulting ROC curves, over the 140 samples drawn over the 14 days. And in Table 2 we show the average corresponding AUC values, and their associated standard errors. We see that for clustering and betweenness on both $G_P$ and $G_W$, the average AUC over 14 days is nearly identical to what we saw for the first day. Particularly, on $G_P$, the average is essentially the same for both clustering (*i.e.,* 0.75 for Day 1 versus 0.744 for 14 days) and betweenness (*i.e.,* 0.724 for Day 1 versus 0.718 for 14 days); and on $G_W$, the average clustering (i.e., 0.787 versus 0.7625) and betweenness (*i.e.,* 0.581 versus 0.562) are also comparable. This suggests a good level of robustness of our results, on average, to the choice of day.

These results also lend support to the conclusion that the clustering metric provides much better discriminatory power than the

**Table 2: Summary of AUC for Detection over 14 Days.**

| | Mean(AUC) | SE(AUC) |
|---|---|---|
| Clustering on $G_P$ | 0.7440 | 0.0103 |
| Betweenness on $G_P$ | 0.7180 | 0.0084 |
| Clustering on $G_W$ | 0.7625 | 0.0080 |
| Betweenness on $G_W$ | 0.5621 | 0.0034 |

betweenness metric. On the other hand, the emerging picture is that the weighted graph $G_W$, despite its more nuanced view of antisocial communication (*i.e.,* through the incorporation of information on multiple flows between source-destination pairs), does not offer a significant improvement in detection accuracy over the simpler, unweighed $G_P$.

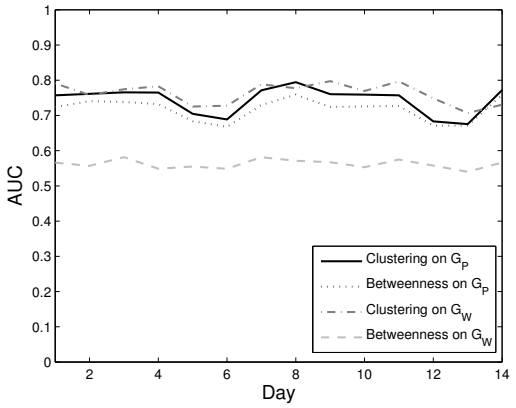


**Figure 7: AUC as a function of day, over 14 days.**

A slightly more refined picture emerges in examining Figure 7, where we show the daily average AUC for each of the four detection methods, with the averages here computed over the 10 sampled flow datasets each day. The associated standard errors (not shown) are on the order of at most 0.03 in all cases. We see that the relative performance of each of the four methods is quite stable with respect to each other over this two-week period. Somewhat interestingly, the performance of the topmost scoring method, based on clustering on $G_P$,while largely constant over the two weeks, clearly degrades slightly in two places, which correspond to weekends (*i.e.,* May 5-6 and 12-13). The other three methods appear to be affected to much lesser extents, or not at all, by the weekend. It is unclear what may be causing this differentiation in relative performance.

# 6. RELATED WORK

As mentioned already, our work is informed by recent studies showing that distinctive, community-based graph structures are evident in normal network traffic [8]. We use these structures to motivate our search for intruders as hosts that do not respect such community boundaries.

A number of other efforts have used properties of the graphs induced by flow records to detect malicious traffic. The notion that malicious hosts have high degree in the bipartite flow graph has motivated algorithms for identifying such "high cardinality" or "superspreader" nodes [9, 11]. However, these approaches rely only on node degree and do not incorporate any notion of graph structures reflecting social properties (*e.g.,* clustering or betweenness). The significant number of malicious hosts in our data that are not "superspreaders" was illustrated in Section 4. The authors in [3] also investigate a graph-based method for intrusion detection. However, their focus is on identifying large scale patterns such as worm outbreaks and their approach is rule-based. Likewise, the authors in [21] consider the same problem and apply simple rules to identify intrusion between clusters of nodes. As with the high-degree heuristic, these methods are not based on social properties

In our work we start from a definition of normality that incorporates social interactions. Other work has considered intrusion detection based on a simpler definition of normality: namely, normality is simply what is common or frequent. This leads to the problem of identifying unusual substructures in graphs [15, 5]. The drawback to this approach is that attacks must be rare in order to be recognized as anomalous. Further, the particular methods used are highly sensitive to noise, which is a significant drawback when applied to sampled flow data.

The work in [14] is focused on worm detection, but has some similarity to our work in that historical communication patterns are used to define communities for each host. However the approach taken is rigid and not based on known properties of social networks, but rather on defining normal communication as that which has happened in the past.

In this paper we build our most detailed view of community-based similarity using *the cosine similarity measure*. Other work has looked at different definitions of community-based similarity for bipartite graphs. For example, [20] defines a similarity-based neighborhood of a particular node by using a random walk with restarts at the node. Our work required a similarity measure between pairs of nodes.

# 7. DISCUSSION

As discussed in the Introduction, signature-based intrusion detection systems are quickly outdated, which has spurred interest on identifying characteristics of normal traffic that are violated by intrusions. Of course, no single characteristic is sufficient for this purpose. While attention has focused on high cardinality hosts, we show that this characteristic alone cannot reliably detect intrusions.

Accordingly, our focus is on new characteristics that can be exploited in the context of a multi-faceted or multi-filter intrusion detection system, to augment approaches based on cardinality and other metrics. Our hypothesis is that communities are exposed in communication graphs, and that community structure can be exploited for intrusion detection. Our results (in particular, showing high AUC values – as high as 79%) show that indeed, communication that consistently crosses community boundaries is often evidence of intrusion. Another point discussed in the Introduction is that validated intrusion data is rare and quickly outdated. In this study we have used logs from the DShield database to identify hosts that are malicious, which is of course only an approximation to ground truth. However, one result of this approximation is that our stated results may be somewhat pessimistic — there may be many malicious sources that are in fact detected by our methods but are not labeled as such because they do not appear in DShield logs. Indeed, there is some evidence that this is the case. For example, in our single-day dataset, we find that 3.1% of detections that are labelled as DShield sources send to destination port 0. This is not surprising, as port 0 is a common attack target. However, we also find that 1.6% of the detections that are not labelled as DShield sources also send to port 0. This is in contrast to the negatives (sources that are not detections), in which only 0.6% send to port 0. The fact that there is a higher tendency to exhibit this characteristic of malicious behavior among even the non-DShield detections

as compared to the negatives suggests that our methods may in fact be performing better than our results indicate.

The limitation of DShield data as a definition of ground truth led us to explore other ways of identifying malicious sources. To this end, we also analyzed a corpus of honeynet logs provided by the Internet Motion Sensor (IMS) project at the University of Michigan [1] collected over the same period in 2007 as the flow logs. The IMS honeynet monitors a large portion of a /8 address space, and report non-spoofed source IP addresses that sent packets to the honeynet. Similar to Dshield, those sources are typically considered to be malicious, but the nature of the data is complementary: DShield is based on signature detection, while the IMS data collects traffic to dark space. Our results using this data were consistent with our DShield results. In particular, the clustering metric applied to $G_P$ showed best performance overall, with a relatively high AUC value (0.63) suggesting that socially-based analysis is effective for detecting these sorts of malicious sources as well.

# 8. CONCLUSION

This paper evaluates a variety of ways of representing and measuring the social information contained in network flow data, and shows how to identify traffic sources that engage in (anti)social behavior — that is, communication that does not respect community structure.

We find that traditional approaches to community identification (*e.g.,* based on modularity) operate at too coarse a level to be useful for this problem. In contrast, a more useful approach is a local one, based on the idea of a cut-vertex. We show that one can make the notion of cut-vertex more robust, in our context of intrusion detection, by using metrics from the social network literature: clustering and betweenness. We find that clustering provides a more effective detector of antisocial behavior than does betweenness, but that the added complexity of the weighted graph $G_W$ does not deliver a corresponding improvement in detection ability over its simpler cousin $G_P$. We also show that the use of social properties is more powerful than the use of source degree (cardinality), in the sense that it allows detection of malicious low-degree sources that can not be detected by the degree-based detector.

Having identified appropriate graph representations and appropriate socially-oriented metrics, we form and evaluate a detection system for malicious sources. Despite the fact that the resulting system operates on flow data (without signature inspection), we show that the resulting system effectively detects the vast majority of sources that the DShield logs identify (using signatures). This holds promise for the use of socially-based methods on data from core networks where many more flows are visible than at network edges (where most intrusion detection systems operate).

## Acknowledgements

We are grateful to Johannes Ullrich and the SANS Institute (`www.sans.org`) for making DShield logs available to us for the purposes of this study. We also thank Michael Bailey and the IMS project members at the University of Michigan for making their honeynet data available to us.

This work was supported in part by NSF grants CNS-0831427, CNS-0905186, CNS-1018266, CNS-1012910, and CNS-1117039. Any opinions, findings, conclusions or other recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the NSF. Partial support was also provided by ONR award N000140910654.

# 9. REFERENCES

[1] M. Bailey, E. Cooke, F. Jahanian, J. Nazario, and D. Watson. The Internet Motion Sensor: A Distributed Blackhole Monitoring System. In *Proceedings of the Network And Distributed Security Symposium*, San Diego, CA, January 2005.

[2] P. Barford, R. Nowak, R. Willett, and V. Yegneswaran. Toward a Model for Sources of Internet Background Radiation. In *Proceedings of the Passive and Active Measurement Conference*, Adelaide, Australia, March 2006.

[3] S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, J. Rowe, S. Staniford-Chen, R. Yip, and D. Zerkle. The Design of GrIDS: A Graph-Based Intrusion Detection System. UC Davis Technical Report CSE-99-2, 1999.

[4] A. Clauset, MEJ Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):66111, 2004.

[5] G. Cormode, F. Korn, S. Muthukrishnan, and Y. Wu. On signatures for communications graphs. In *Proceedings of ICDE*, Cancun, Mexico, April 2008.

[6] W. de Nooy, A. Mrvar, and V. Batagelj. *Exploratory Social Network Analysis with Pajek*. Cambridge University Press, New York, 2005.

[7] Benjamin H. Good, Yves A. de Montjoye, and Aaron Clauset. Performance of modularity maximization in practical contexts. *Physical Review E*, 81(4):046106+, April 2010.

[8] Y. Jin, E. Sharafuddin, and Z. Zhang. Unveiling core network-wide communication patterns through application traffic activity graph decomposition. In *Proceedings of ACM SIGMETRICS*, Seattle, WA, June 2009.

[9] Yu Jin, Jin Cao, Aiyou Chen, Tian Bu, and Zhi-Li Zhang. Identifying high cardinality Internet hosts. In *Proceedings of IEEE INFOCOM*, Rio de Janeiro, Brazil, April 2009.

[10] J. Jung, V. Paxson, A. Berger, and H. Balakrishnan. Fast Portscan Detection Using Sequential Hypothesis Testing. In *IEEE Symposium on Security and Privacy 2004*, Oakland, CA, May 2004.

[11] N. Kamiyama, T. Mori, and R. Kawahara. Simple and adaptive identification of superspreaders by flow sampling. In *Proceedings of IEEE INFOCOM*, Anchorage, AK, April 2007.

[12] KDD Cup data. `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`, August 1999.

[13] Eric D. Kolaczyk. *Statistical Analysis of Network Data: Methods and Models*. Springer, New York, 2009.

[14] P. McDaniel, S. Sen, O. Spatscheck, J. van der Merwe, B. Aiello, and C. Kalmanek. Enterprise security: A community of interest based approach. In *Proceedings of NDSS*, San Diego, CA, February 2006.

[15] C. Noble and D. Cook. Graph-based anomaly detection. In *Proceedings of SIGKDD*, Washington, DC, August 2003.

[16] V. Paxson. Bro: A system for detecting network intruders in real-time. *Computer Networks, (Special Issue on Intrusion Detection)*, 31(23-24):2435–2463, December 1999.

[17] H. Ringberg, A. Soule, J. Rexford, and C. Diot. Sensitivity of pca for traffic anomaly detection. In *Proceedings of ACM SIGMETRICS*, San Diego, CA, June 2007.

[18] M. Roesch. Snort – lightweight intrusion detection for networks. In *Proceedings of the Usenix LISA Conference*, Seattle, WA, November 1999.

[19] S. Rubin, S. Jha, and B. Miller. Automatic generation and analysis of nids attacks. In *Proceedings of Annual Computer Security Applications Conference (ACSAC)*, Tucson, AZ, December 2004.

[20] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. Relevance search and anomaly detection in bipartite graphs. *Proceedings of SIGKDD Explorations Special Issue on Link Mining*, 7(2), 2005.

[21] J. Tolle and O. Niggemann. Supporting Intrusion Detection by Graph Clustering and Graph Drawing. In *Proceedings of the Symposium on Recent Advances in Intrusion Detection*, Toulouse, France, October 2000.

[22] J. Ullrich. The Dshield Project. http://www.sans.org, 2010.

[23] V. Yegneswaran, P. Barford, and J. Ullrich. Internet Intrusions: Global Characteristics and Prevalence. In *Proceedings of ACM SIGMETRICS*, San Diego, CA, June 2003.