Overlapping Communities for Identifying Misbehavior in Network Communications

Farnaz Moradi, Tomas Olovsson, Philippas Tsigas

Chalmers University of Technology, Göteborg, Sweden
{moradi,tomasol,tsigas}@chalmers.se

Abstract. In this paper, we study the problem of identifying misbehaving network communications using community detection algorithms. Recently, it was shown that identifying the communications that do not respect community boundaries is a promising approach for network intrusion detection. However, it was also shown that traditional community detection algorithms are not suitable for this purpose.
In this paper, we propose a novel method for enhancing community detection algorithms, and show that contrary to previous work, they provide a good basis for network misbehavior detection. This enhancement extends disjoint communities identified by these algorithms with a layer of auxiliary communities, so that the boundary nodes can belong to several communities. Although non-misbehaving nodes can naturally be in more than one community, we show that the majority of misbehaving nodes belong to multiple overlapping communities, therefore overlapping community detection algorithms can also be deployed for intrusion detection. Finally, we present a framework for anomaly detection which uses community detection as its basis. The framework allows incorporation of application-specific filters to reduce the false positives induced by community detection algorithms. Our framework is validated using large *email networks* and *flow graphs* created from real network traffic.

1 Introduction

Network intrusion detection systems are widely used for identifying anomalies in network traffic. Anomalies are patterns in network traffic that do not conform to normal behavior. Any change in the network usage behavior, for example caused by malicious activities such as DoS attacks, port scanning, unsolicited traffic, and worm outbreaks, can be seen as anomalies in the traffic.

Recently, it was shown that network intrusions can successfully be detected by examining the network communications that do not respect the community boundaries [9]. In such an approach, normality is defined with respect to social behavior of nodes concerning the communities to which they belong and intrusion is defined as "*entering* communities to which one does not belong".

A community is typically referred to as a group of nodes that are densely interconnected and have fewer connections with the rest of the network. However, there is no consensus on a single definition for a community and a variety of

definitions have been used in the literature [13, 19, 28]. For network intrusion detection, Ding et al. [9] defined a community as a group of source nodes that communicate with at least one common destination. They also showed that a traditional community detection algorithm which is based on a widely used definition, i.e., modularity, is not useful for identifying intruding nodes.

In this paper, we extend and complement the work of Ding et al. [9] by looking into other definitions for communities, and investigate whether the communities identified by different types of algorithms can be used as the basis for anomaly detection. Our hypothesis is that misbehaving nodes tend to *belong to multiple communities*. However, a vast variety of community detection algorithms partition network nodes into disjoint communities where each node only belongs to a single community, therefore they cannot be directly used for verifying our hypothesis. Therefore, we propose a simple novel method which enhances these disjoint communities with a layer of *auxiliary communities*. An auxiliary community is formed over the boundary nodes of neighboring communities, allowing nodes to be members of several communities. This enhancement enables us to show that, in contrary to [9], it is possible to use traditional community detection algorithms for identifying anomalies in network traffic.

In addition to traditional community detection algorithms, another class of algorithms exist which allow a node to belong to several overlapping communities [26]. In this study, we compare a number of such *overlapping algorithms* with our proposed enhancement method for non-overlapping community detection algorithms for network anomaly detection.

Finally, we propose a framework for network misbehavior detection. The framework allows us to incorporate different community detection algorithms for identifying anomalous nodes that belong to multiple communities. However, since legitimate nodes can also belong to several communities [28], application-specific filters can be used for discriminating the legitimate nodes from the anti-social nodes in the community overlaps, thus reducing the induced false positives.

We have evaluated the framework by using it for network intrusion detection and unsolicited email detection in large-scale datasets collected from a high-speed Internet backbone link. These types of misbehavior have traditionally been very hard to detect without inspecting the content of the traffic. To conclude, we show that by using our methodology, it is possible to effectively detect misbehaving traffic by only looking at the network communication patterns.

The remainder of the paper is organized as follows. Section 2 presents related work. Section 3 presents our proposed method for uncovering community overlaps. The framework is presented in Section 4. Section 5 summarizes our findings and experimental results. Finally, Section 6 concludes our work.

2 Related Work

Anomaly detection has been extensively studied in the context of different application domains [6]. In this study, we propose a new graph-based anomaly detection method for identifying network intrusion and unsolicited email in real

network traffic. Although there has been considerable amount of research on detecting these types of misbehavior, it is still a challenge to identify anomalies by merely investigating communication patterns without inspecting their content.

A taxonomy of graph-based anomaly detection methods can be found in [2]. A number of previous studies have proposed methods for finding unusual subgraphs, anomalous substructure patterns, and outlier nodes inside communities in labeled graphs [11, 21, 14]. In this study, we merely use the graph structure and therefore we consider only plain graphs without any labels.

Akoglu et al. [3] proposed a method to assign anomaly scores to nodes based on *egonet* properties in weighted networks. Our framework allows us to incorporate such properties as application-specific filters. Sun et al. [25] proposed a method for identifying anomalous nodes that are connected to irrelevant neighborhoods in bipartite graphs. Ding et al. [9] showed that although finding the cut-vertices can be used for intrusion detection, more robust results can be achieved by using clustering coefficient in a one-mode projection of a bipartite network. Moreover, they showed that using a modularity maximization community detection algorithm [7] is not suitable for spotting network intruders.

In this paper, we revisit the problem of finding anomalous nodes in bipartite/unipartite plain graphs by using community detection algorithms. We deploy an alternative definition for an anomaly as suggested in [9] and confirm their finding that maximizing modularity is not suitable for identifying intruders on its own. However, we show that there are several types of algorithms which are useful for misbehavior detection if enhanced with auxiliary communities.

3 Community Detection

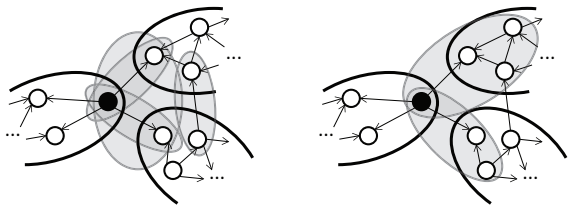
In this section, we introduce a novel approach which enables us to deploy existing community detection algorithms for identifying anomalies in network traffic.

3.1 Auxiliary Communities

In this paper, we introduce the concept of auxiliary communities. An auxiliary community is added over the boundary nodes of disjoint communities, forcing nodes to become members of more than one community.

The most basic approach is to introduce one auxiliary community for each *boundary edge* between two different communities. However, a *boundary node* can have multiple boundary edges. Therefore, an improvement over the above approach is to add only one auxiliary community over a boundary node and all its boundary edges, covering all its neighbors that are members of other external communities (Algorithm 1). Our approach can be further refined to consider the whole one-step neighborhood, i.e., *egonet*, of a boundary node as an auxiliary community instead of just its boundary neighbors.

Ding et al. [9] defined a community in a directed bipartite network as a group of source nodes that have communicated with at least one common destination. In a bipartite network, there are two distinct sets of source nodes and destination



(a) Neighboring (b) Egonet of sinks

Fig. 1: Auxiliary communities.

nodes. Based on this definition, the source nodes that belong to the egonet of a destination node form a community. In a unipartite network, a distinct set of source and destination nodes does not exist. Therefore, we apply the above definition of communities only to the sink nodes which have only incoming edges (Algorithm 2).

Algorithm 1: Neighboring Auxiliary Communities (NA)

Input: a graph $G(V, E)$; a non-overlapping community set $\mathcal{C}$;
Output: auxiliary community set $\mathcal{A}$;

```
1.  for all v ∈ V do
2.      Com(v) = {C ∈ C : v ∈ V(C)};
3.      for all u ∈ Neighbors(v) do
4.          if Com(v) ≠ Com(u) then
5.              A ← A ∪ {u, v};
6.          end if
7.      end for
8.      A ← A ∪ A;
9.  end for
10. return A
```

Algorithm 2: Egonet Auxiliary Communities of Sinks (EA)

Input: a graph $G(V, E)$; a non-overlapping community set $\mathcal{C}$;
Output: auxiliary community set $\mathcal{A}$;

```
1.  for all v ∈ V do
2.      Com(v) = {C ∈ C : v ∈ V(C)};
3.      for all u ∈ Neighbors(v) do
4.          if Com(v) ≠ Com(u) and
            Sink(u) then
5.              A ← Egonet(u);
6.              A ← A ∪ A;
7.          end if
8.      end for
9.  end for
10. return A
```

Figure 1 shows a comparison of the proposed methods for adding auxiliary communities. It can be seen that each approach places the intruding node (black node) in different auxiliary communities (grey communities). The main difference of our methods is that Algorithm 1 only adds neighboring auxiliary (NA) communities over the boundary nodes, whereas Algorithm 2 also allows the neighbors of the boundary sink nodes to be covered by egonet auxiliary (EA) communities. Therefore, a misbehaving node which is not in the boundary of its community can still belong to multiple communities by using Algorithm 2.

The complexity of adding auxiliary communities for a network with a degree distribution $p_k = k^{-\alpha}$, is $O(nk_{max}^{3-\alpha})$, where n is the number of nodes, k_{max} is the highest degree, and α is the exponent of the degree distribution.

Table 1: Community detection algorithms.
n and m denote the number of nodes and edges, respectively, k_{max} is the maximum degree, t is the number of iterations, and α is the exponent of the degree distribution.

Algorithm	Complexity	
Overlapping	*LC* [1]	$O(nk_{max}^2)$
	LG [12]	$O(nm^2)$
	SLPA [27]	$O(tm)$
	OSLOM [18]	$O(n^2)$
	DEMON [8]	$O(nk_{max}^{3-\alpha})$
Non-Overlapping	*Blondel* (also known as Louvain method) [5]	$O(m)$
	Infomap [22]	$O(m)$
Auxiliary	*NA* (Neighboring Auxiliary Communities)	$O(nk_{max}^{3-\alpha})$
	EA (Egonet Auxiliary Communities)	$O(nk_{max}^{3-\alpha})$

3.2 Community Detection Algorithms

In this paper, we use a number of well-known and computationally efficient (overlapping) community detection algorithms, which are listed in Table 1. Our goal is to investigate which definition of a community and which types of algorithms are most suitable for network misbehavior detection.

LC and LG find overlapping communities in a graph based on the edges. LG, induces a *line graph* from the original network to which any non-overlapping algorithm can be applied. In this paper, we uses a weighted line graph with self-loops, E, and refer to LG using this graph as *LG(E)*. SLPA and OSLOM are both node-based methods and have very good performance [26]. Finally, DEMON is an state-of-the-art node-based, local, overlapping community detection algorithm.

The non-overlapping algorithms used in this study also have very good performance [17]. Blondel greedily maximizes modularity and unfolds a hierarchical community structure with increasing coarseness. In this study, we consider the communities identified at both the last and the first level of the hierarchy and refer to them as *Blondel* and *Blondel L1*, respectively. We also use the communities formed by Blondel as input to OSLOM, which modifies these communities in order to improve their statistical significance. Finally, Blondel L1 is also used to partition the nodes in the induced line graphs by LG(E).

4 Framework

This section presents our framework for community-based anomaly detection. Algorithm 3 shows the first component of our framework, where overlapping algorithms can be directly used, but non-overlapping algorithms only after being enhanced with auxiliary communities.

The second component of our framework consists of a set of graph properties which are used as *filters*. Our hypothesis is that intruding nodes are likely to be placed in community overlaps. However, non-misbehaving nodes can also belong to more than one community, and basing detection merely on community overlaps, can lead to false positives. Therefore, these filters are used to reduce the induced false positives by the community detection algorithms.

Algorithm 3: Community-based anomaly detection

Input: a graph $G(V, E)$; a community detection algorithm CD;
Output: a set AS of $\langle v, score(v)\rangle$;

1. Set $AS = \emptyset$; Set $\mathcal{C} = \emptyset$; Set $\mathcal{A} = \emptyset$;
2. $\mathcal{C} = CD(G)$;
3. **if** CD is non-overlapping **then**
4. $\quad \mathcal{A} \leftarrow Auxiliary(G, \mathcal{C})$;
5. $\quad \mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{A}$;
6. **end if**
7. **for all** $v \in V$ **do**
8. $\quad score(v) \leftarrow Filters(v, G, \mathcal{C})$;
9. $\quad AS \leftarrow \langle v, score(v)\rangle$;
10. **end for**
11. **return** AS

Algorithm 4: Application-specific filters

Input: a node v; a graph $G(V, E)$; a set of communities $\mathcal{C}$; weights $w_i \in [0, 1]$ s.t. $\sum w_i = 1$; user-defined threshold values t_i, where i is the index of the property;
Output: an anomaly score $score(v)$;

1. $Coms(v) = \{C \in \mathcal{C} : v \in V(C)\}$;
2. $\phi_1(v) = |Coms(v)|$;
3. $\phi_2(v) = |Coms(v)|/|Neighbors(v)|$;
4. $\phi_3(v) = 1 - ClusteringCoeff(v)$;
5. $\phi_4(v) = OutDeg(v)/Deg(v)$;
6. $\phi_5(v) = Deg(v)/EdgeWeights(v)$;
7. $score(v) = \sum w_{i\mathcal{I}}(\phi_i(v), t_i)$;
8. **return** $score(v)$

The framework uses a simple method for combining the extracted properties. For each node v in the graph, the anomaly score is calculated as $score(v) = \sum_i w_{i\mathcal{I}}(\phi_i(v), t_i)$, where i is the index of the property which is being aggregated, w_i is a weight for property ϕ_i where $\sum w_i = 1$, and $_{\mathcal{I}}(\phi_i(v), t_i)$ is an indicator function which compares the value of a graph property $\phi_i(v)$ to a corresponding threshold value t_i such that $_{\mathcal{I}}(\phi_i(v), t_i) = \begin{cases} 1, & \phi_i(v) > t_i \\ 0, & \text{otherwise.} \end{cases}$

The threshold values and weights are dependent on the type of data and prior knowledge of normal behavior, which is necessary for anomaly detection and can be achieved from studies of anomaly-free data. Finally, the anomaly score $score(v)$ can be used to quantify to what extent a node v is anomalous.

The properties presented in Algorithm 4 are examples of community and neighborhood properties that we have used as filters in our experiments for intrusion and unsolicited email detection. The selection of appropriate filters depends on the application of anomaly detection.

Network intruders are normally not aware of the community structure of the network, and therefore communicate to random nodes in the network [23]. It is expected to be very expensive for attackers to identify the network communities, and even if they do, limiting their communication with the members in the same community can inversely affect their gain. Therefore, the number of communities per node, as well as the ratio of the number of communities per node over the number of its neighbors, which correspond to ϕ_1 and ϕ_2 in Algorithm 4, respectively, are expected to be promising properties for finding intruders.

The rest of the properties, are graph metrics that correspond to the social behavior of nodes and can be extracted from the direct neighborhood of the nodes. We have used these properties for detecting unsolicited email (Section 5.3) and therefore in the following we explain them in the context of spam detection.

The *clustering coefficient* of a node is known to have a lower value for spammers than legitimate nodes [15, 20]. Property ϕ_3 calculates one minus the clustering coefficient so that spammers are assigned higher values. It has also been shown that spammers are mostly using randomized fake source addresses and therefore it is not expected that they receive many emails [16]. Property ϕ_4 calculates the ratio of the out-degree over the degree of the nodes, which is expected to be high for the spammers. Finally, it has been shown that spammers tend to use the fake source email address to send only a few spam, and target each receiving email address only once [16]. Therefore, the degree of a node over its edge weights, property ϕ_5, is expected to be higher for spammers than legitimate nodes, where the edge weights correspond to the number of exchanged emails.

5 Experimental Results

We have evaluated the usability of different algorithms in our framework using two different datasets which were generated from network traffic collected on a 10 Gbps Internet backbone link of a large national university network.

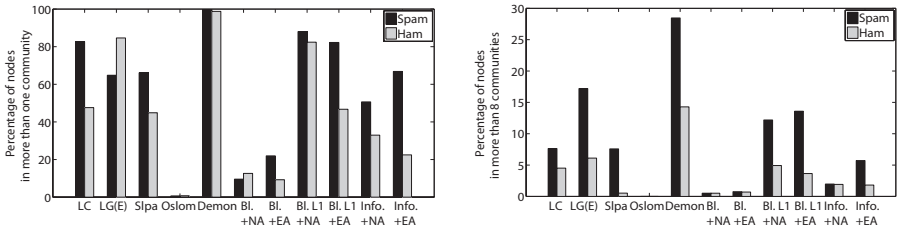
Flow Dataset. The flow level data was collected from the incoming network traffic once a week during 24 hours for seven weeks in 2010 [4]. The flows were used to generate bipartite networks where source and destination IP addresses form the two node sets. The malicious source addresses in the dataset were taken from the lists reported by DShield and SRI during the data collection period [10, 24]. This dataset is used to compare our approach with the method proposed by Ding et al. [9] for network intrusion detection. The datasets are similar with respect to the ground truth and only differ with respect to the collection location and the sampling method used.

Email Dataset. This dataset is generated from captured SMTP packets in both directions of the backbone link. The collection was performed twice (2010 and 2011), where the duration of each collection was 14 consecutive days. This dataset was used for generating *email networks*, in which email addresses represent the nodes, and the exchanged emails represent the edges. The ground truth was obtained from a well-trained content-based filtering tool[1] which classified each email as legitimate (*ham*) or unsolicited (*spam*).

5.1 Comparison of Algorithms

In this section, we present a comparison of the algorithms using the email dataset. Figure 2a shows the percentage of ham and spam nodes (averaged over the 14 days in 2010), which are placed in multiple communities by different algorithms. It can be seen that many ham nodes belong to more than one community, which is an expected social behavior. It can also be seen that, most algorithms place the majority of spammers into more than one community, except OSLOM and Blondel which tend to form very coarse-grained communities.

[1] SpamAssassin (`http://spamassassin.apache.org`) which provided us with an estimated false positive rate of less than 0.1% and a detection rate of 91.4%



(a) percentage of nodes in more than one community

(b) Percentage of nodes in more than 8 communities

Fig. 2: Percentage of nodes in multiple communities in email dataset (2010).

The figure also shows that, regardless of which non-overlapping algorithm being used, adding egonet auxiliary communities (Algorithm 2) places more spam than ham nodes into several communities compared to adding neighboring auxiliary communities (Algorithm 1). The reason is that NA communities are only added over the boundary nodes, however, EA communities also allow the neighbors of the boundary sink nodes to be covered by auxiliary communities.

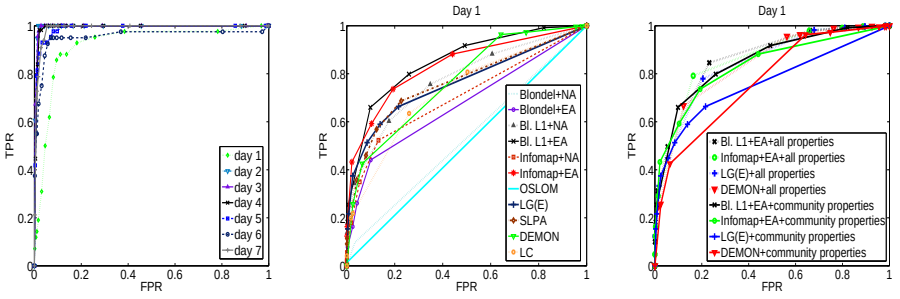
Finally, Figure 2b shows that a higher percentage of spammers belong to more than eight communities compared to legitimate nodes. The same observation holds for the data collected in 2011. Therefore, we can confirm that both fine-grained algorithms enhanced with EA communities, and overlapping algorithms can be used to spot misbehaving nodes based on the number communities to which they belong.

5.2 Network Intrusion Detection

It has been shown that a non-overlapping community detection algorithm (which maximizes modularity) is not suitable for identifying intruders in network flow data [9]. In this study, we have further investigated the possibility of using different community detection algorithms, including a modularity-based one, by using auxiliary communities for network intrusion detection.

One example of network intrusion is port scanning, where a scanner searches for open/vulnerable services on selected hosts. Current intrusion detection systems are quite successful in identifying scanners. In this paper, we just verify the possibility of detecting scanners using a community-based technique.

We generated one bipartite graph from the flows collected for each day. As an example, the flow graph generated from the first day of data contained 51,720 source nodes sending 93,113 flows to 32,855 destination nodes. This includes 607 malicious nodes (based on DShield/SRI reports) that have sent 7,861 flows. We made the assumption that the malicious source nodes that have tried to communicate with more than 50 distinct destinations are suspected of scanning. Figure 3a shows the ROC curves for seven different days. These curves show the trade-off between the true positive rate (TPR) and the false positive rate (FPR). We have used Blondel L1 enhanced with egonet auxiliary communities (EA), and have only used property ϕ_1, i.e., the number of communities to which a node



(a) Scanner detection using community properties (2010)

(b) Spam detection using community properties (2010)

(c) Spam detection using neighbor and community properties (2010)

Fig. 3: Performance of different algorithms for network misbehavior detection.

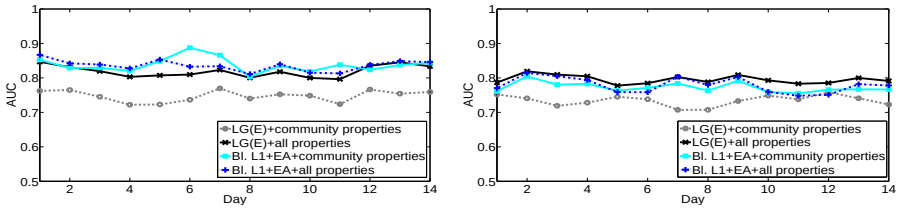
belongs, as the filter. It can be seen that this approach yields high performance with mean area under curve (AUC) of 0.98, where around 90% to 100% of malicious scanners are detected with a FPR of less than 0.05. This observation confirms that our framework is successful in identifying scanners.

Network intrusion attacks are not limited to scanning attacks, therefore we have also tried to identify other malicious (DShield/SRI) sources and have compared our approach with the method proposed by Ding et al. [9]. Our experiments show that the performance of both methods are quite consistent with mean AUC 0.60 (standard error 0.009) for the method by Ding et al. and 0.62 (standard error 0.015) for our approach using LG(E) as the overlapping community detection and properties ϕ_1 and ϕ_2 as filters. Overall, these results confirm that the community structure of a network provides a good basis for network intrusion detection and both non-overlapping communities enhanced with EA communities and overlapping communities can indeed be used for this purpose.

5.3 Unsolicited Email Detection

Our experimental comparison of community detection algorithms in Section 5.1 showed that most of the studied algorithms place spammers into multiple communities. In this section, we investigate how these algorithms can be used in our framework to detect these spammers only by observing communication patterns.

For this study, we have generated one email network from the emails collected for each day. The community detection algorithms were applied to the undirected and unweighted giant connected component of each email network. The edge directions and weights were later taken into account for adding auxiliary communities and calculating different graph properties. We consider an email address to be a spammer if it has sent more than one spam to more than one recipient. As an example, the email network generated from the first day of data in 2010, contains 167,329 nodes and 236,673 edges, where 23,628 nodes were spammers sending 126,145 spam emails. It is important to note that the



(a) 2010 (b) 2011

Fig. 4: Area under the ROC curve for spam detection over time.

vast majority of the spammers have not sent large volumes of email and therefore a simple volume-based detection method would not be suitable for spammer detection.

Figure 3b shows the ROC curves for our spam detection method using different algorithms and the community-based properties ϕ_1 and ϕ_2. It can be seen that OSLOM, which aims at forming statistically significant communities fails to identify spamming nodes. It can also be seen that a node-based overlapping algorithm, SLPA, and an edge-based algorithm, LG(E), perform similarly, and the AUC (not shown in the figure) is identical for both algorithms (0.76).

Figure 3b also shows the ROC curves for non-overlapping algorithms which are enhanced with our auxiliary communities. It can be seen that Blondel, which aims at optimizing modularity, performs very poor. This observation is in accord with the observation in [9] that a modularity maximization algorithm is not suitable for anomaly detection due to its resolution limit. However, Blondel L1 (first level in the community hierarchy of Blondel), which forms finer granularity communities, performs dramatically better than its last level using either type of the auxiliary communities. Moreover, it can be seen that adding EA communities leads to better results compared to NA communities.

Overall, our experiments for different days in both email datasets showed that Blondel L1 and Infomap enhanced with EA, SLPA, LG(E), and DEMON all perform well with respect to placing spamming nodes into multiple communities. In practice, low false positive rates are essential for spam detection, therefore both Blondel L1 with EA communities and LG(E) that allow us to, on average, detect more than 25% and 20% of spamming nodes, respectively, for different days with very low FPR (less than 0.01) are the most suitable algorithms.

These results confirm that our method for adding EA communities to enhance non-overlapping algorithms yields not only comparable, but even better, results than an overlapping algorithm. Although both Blondel L1 with EA communities and LG(E) use the same modularity-based algorithm as their basis (we have applied Blondel L1 on the induced line graph of LG(E)), adding EA communities has also a lower complexity than inducing weighted line graphs (Table 1).

As mentioned earlier, our framework allows us to incorporate a number of application-specific filters to reduce the induced false positives (Algorithm 4). Figure 3c shows a comparison of the spam detection using filters based on community properties (ϕ_1 and ϕ_2 only) and the combination of community and

neighborhood properties (ϕ_1 - ϕ_5) for the first day of data in 2010. It can be seen that use of additional filters improves the detection (the same observation also holds for the algorithms not shown).

Finally, Figure 4 shows the AUC for spam detection using our framework with LG(E) and Blondel L1 enhanced with EA communities over 14 days during 2010 and 2011. It can be seen that the results are quite stable over time and the AUC of our method for adding EA communities compared to a more complex overlapping algorithm is much better when only community properties are used.

6 Conclusions

In this paper, we have evaluated the performance of community detection algorithms for identifying misbehavior in network communications. This paper extends and complements the previous work on community-based intrusion detection, by investigating a variety of definitions for a community, introducing auxiliary communities for enhancing traditional community detection algorithms, and showing that, in contrary to previous work, these algorithms can indeed be deployed as the basis for network anomaly detection.

We have also provided a framework for community-based anomaly detection which allows us to find the nodes that belong to multiple communities by either using auxiliary communities or overlapping algorithms. It also enables us to deploy neighborhood properties, which are indicative of social behavior, for discriminating the nodes that naturally belong to more than one community from the anti-social ones. The applicability of our framework for identifying network intrusions and unsolicited emails was evaluated using two different datasets coming from traffic captured on an Internet backbone link. Our experiments show that our framework is quite effective and provides a consistent performance over time. These results suggest that detecting community overlaps is a promising approach for identifying misbehaving network communications.

Acknowledgments

This work was supported by .SE – The Internet Infrastructure Foundation and SUNET. The research leading to these results has also received funding from the European Union Seventh Framework Programme (FP7/ 2007-2013) under grant agreement no. 257007.

References

1. Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 466(7307):761–4, Aug. 2010.
2. L. Akoglu and C. Faloutsos. Anomaly, event, and fraud detection in large network datasets. In *WSDM*, page 773. ACM Press, 2013.
3. L. Akoglu and M. McGlohon. Oddball: Spotting Anomalies in Weighted Graphs. In *PAKDD*, pages 410–421, 2010.

4. M. Almgren and W. John. Tracking Malicious Hosts on a 10Gbps Backbone Link. In *15th Nordic Conference in Secure IT Systems*, 2010.
5. V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast Unfolding of Communities in Large Networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, Oct. 2008.
6. V. Chandola, A. Banerjee, and V. Kumar. Anomaly Detection: A Survey. *ACM Computing Surveys*, 41(Sep):1–72, 2009.
7. A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical review. E*, 70(6 Pt 2):066111, Dec. 2004.
8. M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi. DEMON: a local-first discovery method for overlapping communities. In *ACM SIGKDD*, page 615, 2012.
9. Q. Ding, N. Katenka, P. Barford, E. Kolaczyk, and M. Crovella. Intrusion as (anti)social communication. In *ACM SIGKDD*, page 886, 2012.
10. DShield. Recommended block list, http://www.dshield.org/block.txt, 2010.
11. W. Eberle and L. Holder. Anomaly detection in data represented as graphs. *Intelligent Data Analysis*, 11(6):663–689, 2007.
12. T. Evans and R. Lambiotte. Line graphs, link partitions, and overlapping communities. *Physical Review E*, 80(1):1–8, July 2009.
13. S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, Feb. 2010.
14. J. Gao, F. Liang, W. Fan, C. Wang, Y. Sun, and J. Han. On community outliers and their efficient detection in information networks. In *ACM SIGKDD*, 2010.
15. L. Gomes, R. Almeida, and L. Bettencourt. Comparative Graph Theoretical Characterization of Networks of Spam and Legitimate Email. In *CEAS*, 2005.
16. C. Kreibich, C. Kanich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage. On the Spam Campaign Trail. In *LEET*, pages 697–8, 2008.
17. A. Lancichinetti and S. Fortunato. Community Detection Algorithms: A Comparative Analysis. *Physical Review E*, 80(5):1–11, Nov. 2009.
18. A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato. Finding statistically significant communities in networks. *PloS one*, 6(4):e18961, Jan. 2011.
19. J. Leskovec, K. J. Lang, and M. Mahoney. Empirical Comparison of Algorithms for Network Community Detection. In *WWW*, page 631, 2010.
20. F. Moradi, T. Olovsson, and P. Tsigas. Towards modeling legitimate and unsolicited email traffic using social network properties. In *SNS*, 2012.
21. C. C. Noble and D. J. Cook. Graph-based anomaly detection. In *ACM SIGKDD*, pages 631–636, 2003.
22. M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *National Academy of Sci.*, 105(4):1118–23, Jan. 2008.
23. N. Shrivastava, A. Majumder, and R. Rastogi. Mining (Social) Network Graphs to Detect Random Link Attacks. In *ICDE*, pages 486–495. IEEE, 2008.
24. SRI. International Malware Threat Center, most aggressive malware attack source and filters, `http://mtc.sri.com/live_data/attackers/`, 2010.
25. J. Sun, D. Qu, Huiming Chakrabarti, and C. Faloutsos. Neighborhood Formation and Anomaly Detection in Bipartite Graphs. In *ICDM*, pages 418–425, 2005.
26. J. Xie, S. Kelley, and B. Szymanski. Overlapping community detection in networks: the state of the art and comparative study. *ACM Computing Surveys*, 45(4), 2013.
27. J. Xie and B. Szymanski. Towards Linear Time Overlapping Community Detection in Social Networks. In *PAKDD*, pages 25–36. Springer-Verlag, 2012.
28. J. Yang and J. Leskovec. Defining and evaluating network communities based on ground-truth. In *ICDM*, page 745754. IEEE, 2012.