CoVerD: Community-based Vertex Defense against Crawling Adversaries

Pegah Hozhabrierdi and Sucheta Soundarajan

EECS Department, Syracuse University, Syracuse NY, USA,
{phozhabr,susounda}@syr.edu

Abstract. The problem of hiding a node inside of a network in the presence of an unauthorized crawler is shown to be NP-complete. The available heuristics tackle this problem from mainly two perspectives: (1) the local immediate neighborhood of the target node (local perturbation models) and (2) the global structure of the graph (global perturbation models). While the objective of both is similar (i.e., decreasing the centrality of the target node), they vary substantially in their performance and efficiency; the global measures are computationally inefficient in the real-world scenarios, and the local perturbation methods deal with the problem of constrained performance. In this study, we propose a community-based heuristic, *CoVerD*, that retains both the computational efficiency of local methods and the superior performance of global methods in minimizing the target's closeness centrality. Our experiments on five real-world networks show a significant increase in performance by using *CoVerD* against both BFS and DFS crawling attacks. In some instances, our algorithm successfully increased the crawler's budget by 3 and 10 times compared to the next best-performing benchmark. The results of this study show the importance of the local community structure in preserving the privacy of the nodes in a network, and pave a promising path for designing scalable and effective network protection models.

Keywords: community, protection, crawling, closeness centrality

1 Introduction

With the increase in digitization of entities and their data, the problem of maintaining the privacy of critical nodes in a network has become ever more relevant [22]. Many versions of this problem have been considered. For example, Waniek et al. examine how to protect individuals from detection in a crawling attack [25], and others have addressed the susceptibility of easily accessible public profiles in de-anonymization attempts [10, 18, 21, 24]. Numerous studies have been dedicated to finding the optimal crawling strategy for network adversarial attacks and methods for their timely detection [2, 3, 5, 15, 23]. This problem can be considered from a variety of angles, including designing effective attack strategies, early detection of attacks, or network defense, in which the goal is to minimally perturb the network so as to protect the target nodes from detection.

In this work, we consider the problem of network defense. We assume that the defender has a limited *defense budget* (i.e., number of allowable edge perturbations) for protecting the target nodes, and has no knowledge of the attacker's logistics (i.e., starting point and the crawling algorithm). Note that in this problem, the highest level of protection is achieved by isolating the target. However, this is not an acceptable solution, as these targets are likely to be important to the network, and so removing their connections harms the functionality of the system. The literature contains few works dealing with network protection strategies from the defender's perspective. Previous work has shown that target node protection from the defender's perspective results is NP-complete [9, 16]. Heuristic solutions use either global graph perturbations [4, 14] or local graph perturbations [1, 25]. The time complexity of the former is substantially greater, and often do not substantially outperform the local methods. However, the search space for local-based methods is small and they rapidly reach their performance plateau regardless of budget (see Section 4).

In this study, we propose a new approach for vertex defense against crawling attacks: *community-based local graph perturbations*, which find a middle ground between the fast computation of local perturbations and larger search space of global methods.[1] The only information required by *CoVerD* is the community labels of nodes and their 1-hop neighborhood, which is the same information required by local network perturbation heuristics [25]. Therefore, our algorithm is fast and, due to its budget-aware decision making, surpasses the performance of both local and global perturbation heuristics (see Section 5). In fact, we show that *CoVerD* has the same impact on reducing the closeness centrality of the target node without the need for expensive computation of centrality. The summary of **our contributions** is as follows.

- We formulate the problem of node protection in complex networks from a defender's perspective. We consider the general case in which the defender has no information on the attacker's starting point or its crawling algorithm.
- We propose a more general heuristic which considers both the local and community information of the target node. Our community-based defender, *CoVerD*, is fast and benefits from the advantages of local network perturbations (namely, computational cost and defending budget) while bypassing its shortcomings (namely, the rapid performance plateau). *CoVerD* can achieve close to optimal performance (i.e., the attacker's budget is maximized) and effectively reduces the closeness centrality of the target node.
- On five real-world networks of varying sizes, we show the superiority *CoVerD* in terms of efficiency and performance against different crawling adversaries.[2]

2 Related Work

The problem of defending target nodes against crawling attacks has been studied in four general domains: (1) optimization of crawling techniques for data acqui-

[1] 'Community' here refers to a topological community.
[2] The source code will be made available upon publication.

sition (attacker's perspective) [10, 18, 21, 24]; (2) detection of malicious crawling behavior (attacker's perspective) [17]; (3) increasing network robustness against crawling attack through global network perturbations (defender's perspective) [9, 14, 16]; (4) protecting target nodes through local network perturbations (defender's perspective) [1, 25]. As our focus is on defender's perspective, here we only discuss the latter two categories in depth.

2.1 Defense via Local Network Perturbations

A simple, yet effective, method in locally manipulating graph structure is ROAM (Remove One, Add Many), the algorithm proposed by [25]. ROAM follows the intuition that the most important factor in the target node's exposure is its immediate neighborhood. ROAM decreases the degree centrality of the target node by iteratively removing its highest-degree neighbors and connecting them to other immediate neighbors of the target node, ensuring that the average path length and connectivity in the target's neighborhood are preserved. Abrahamsson adopts the same algorithm but uses eigenvector centrality to pick the neighbor candidate [1]. Their results are comparable to that of ROAM, but incur greater computational costs. The main drawback of ROAM is the limit to its performance. Once the target node's degree reaches 1, ROAM stops and the algorithm reaches its plateau even in the presence of more protection budget. Our method matches or beats ROAM's performance for small budgets, but rapidly reaches close to optimal performance with a slight increase in budget.

2.2 Defense via Global Network Perturbations

The majority of works in this category use various edge and/or node centralities to greedily remove edges. The objective is to minimize/maximize a global network measure, such as network centrality or average path length. Crescenzi et al. address the complementary problem to ours: maximizing the visibility of a node in the network [9]. They achieve this goal by greedily adding outgoing edges from the target node such that the closeness or betweenness centrality of the target node is maximized. Their time complexity is $\mathcal{O}(k.n.g(n,m))$, where n and m are the number of nodes and edges respectively, and $g(n,m)$ is the complexity of computing either either closeness or betweenness centrality for a node in the graph. Numerous works have proposed methods to reduce the complexity of $g(n,m)$ [6, 8, 19, 20], among which Ji et al. [14] specifically tailored their method for the vertex protection problem. The time complexity of their approach is $\mathcal{O}(k \cdot m \cdot \tau_{mn})$, in which τ_{mn} represents the number of traverse nodes and edges that can be computed in $\mathcal{O}(m+n)$ in the worst case. Despite these efforts, the greedy approach using global network measures do not outperform local measures, such as ROAM [4]) and are infeasible on large scale real-world networks. As such, a few studies have used greedy removal of edges without re-computation of the centrality measures as well[14, 16]. We use three of these methods as baselines that have substantially higher computational cost and worse performance than local methods, including our algorithm, *CoVerD*.

Algorithm 2: `CoVerD`

```
Input: $G$, $t$, $\mathfrak{C}(t)$, $b_d$
$continue \leftarrow True$
while $continue$ do
    $G, b_d, continue \leftarrow$ IncreaseTargetLoyalty($G$, $t$, $\mathfrak{C}(t)$, $b_d$)
    $N \leftarrow \mathcal{N}_{\mathcal{G}_t}(t)$
    $G, b_d, continue \leftarrow$ ROAM($G$, $t$, $b_d$)
    $G, b_d, continue \leftarrow$ Increase1HopLoyalty($G$, $N$, $t$, $\mathfrak{C}(t)$, $b_d$)
    $G, b_d, continue \leftarrow$ BuildLoyaltyChamber($G$, $t$, $\mathfrak{C}(t)$, $b_d$)
end
return $G$
```

3 Preliminaries and Problem Definition

Network Notation. Let $G = (V, E)$ ($|V| = n$ and $|E| = m$) be a connected, undirected, unweighted graph with a mapping $\mathcal{C}(.)$ that projects $|V|$ onto non-overlapping partitions. For each node $t \in V$, we represent its 1-hop (immediate) neighborhood in G as $\mathcal{N}_G(t) = \{j | j \in V, (t, j) \in E\}$ and its cohort as $\mathfrak{C}(t) = \{j | j \in V, \mathcal{C}(j) = \mathcal{C}(t)\}$. The subgraph of G that contains the nodes in $\mathfrak{C}(t)$ is denoted as $\mathcal{G}_t = (\mathfrak{C}(t), \mathcal{E}_t)$, in which $\mathcal{E}_t = \{(i, j) | i, j \in \mathfrak{C}(t), (i, j) \in E\}$. We will refer to this induced subgraph as the node's *cohort subgraph*. Also, the connectivity of the cohort subgraph is the only necessary condition for our algorithm and we can generalize our approach to directed and/or weighted graphs as well (see 4.1).

Algorithm 1: `ROAM`

```
Input: $G$, $t$, $b_d$
$flag \leftarrow False$
$spent \leftarrow 0$
while $spent \leq b_d$ do
    $\mathcal{N}_{\mathcal{G}_t}(t) \leftarrow$ SortByDegree($\mathcal{N}_{\mathcal{G}_t}(t)$)
    for $p \in \mathcal{N}_{\mathcal{G}_t}(t)$ do
        $G' \leftarrow G(V, E \setminus \{(t, p)\})$
        if $IsConnected(G')$ then
            $G \leftarrow G'$
            $q \leftarrow$Random($\mathcal{N}_{\mathcal{G}_t}(t)$)
            $G \leftarrow G(V, E \cup \{(p, q)\})$
            $spent \leftarrow spent + 2$
        end
    end
end
$b_d \leftarrow b_d - spent$ if $b_d \leq 0$ then
    $flag \leftarrow True$
end
return $G, b_d, flag$
```

Problem Definition. The vertex defense problem (also referred to as *node protection* and *hiding node* problem [16, 25]) involves a target node t and two actors: a crawling adversary (attacker) and a defender. If we denote the crawling algorithm used by the attacker as $\mathcal{A}$ and the probability of $\mathcal{A}$ visiting a node u in G at the l^{th} step as $P_{\mathcal{A}}(u, G, l)$, the adversary's objective is to find an optimal $\mathcal{A}^*$ such that $\mathcal{A}^* = \max_{l, \mathcal{A}, b_a} P_{\mathcal{A}}(t, G, l)$ for $l \leq b_a$ and $b_a \ll n$.

The defender has no knowledge of the attacker's logistics (crawling algorithm, budget, or starting point). It only has information on the target node and the community it belongs to. Within a limited budget $b_d \ll m$, the defender has the ability to perturb any set of edges in the community of t to obtain a new graph $G^{'}$. Its objective is to find the optimal perturbed graph G^* such that $G^* = \min_{l, G', b_a} P_{\mathcal{A}}(t, G^{'}, l)]$ for $l \leq b_a$ and $b_a \ll n$.

4 Method

The goal of *CoVerD* is to minimize the closeness centrality of the target node (which increases the attacker's required budget) as much as the available defense budget permits, by using only the information of the cohort neighborhood of the target node. This strategy is intuitive: focus on the immediate neighborhood of the target for small budgets and expand attention to further neighborhoods within the cohort as the budget increases. To this end, *CoVerD* uses a hierarchical modular structure to achieve the proper distribution of the available budget.

There are two underlying assumptions behind *CoVerD*'s intuition: (1) for decreasing the centrality of a node, its 1-hop neighborhood plays a more prominent role than its larger k-hop neighborhood; (2) the existence of a *protective* community structure (i.e., with high average loyalty score) around the target node overpowers the global pathways to the target node. The first assumption is already shown to be the case in real-world social networks [4, 25]. In this study, we intend to show that the second assumption holds for these networks.

Algorithm 3: `IncreaseTargetLoyalty`

```
Input: $G$, $t$, $\mathfrak{C}(t)$, $b_d$
$flag \leftarrow True$
$spent \leftarrow 0$
$N' \leftarrow$ SortByDegree($\mathcal{N}_G(t) \setminus \mathcal{N}_{\mathcal{G}_t}(t)$)
for $p \in N'$ do
    $G' \leftarrow G(V, E \setminus \{(t,p)\})$
    if IsConnected($G'$) then
        $G \leftarrow G'$
        $spent \leftarrow spent + 1$
        if $spent > b_d$ then
            break
        end
    end
end
$b_d \leftarrow b_d - spent$
if $b_d \leq 0$ then
    $flag \leftarrow False$
end
return $G, b_d, flag$
```

Cohort Loyalty Score. While it is tempting to fully isolate a cohort containing a sensitive node, in practice, outgoing connections from cohort are necessary to keep the functionality of the network, even though they increase the cohort's vulnerability. As such, we assign a loyalty score to each node inside of the cohort to signify the impact that a node in the cohort has on exposing the cohort to the rest of the network. Formally, for each node i inside of a cohort $\mathfrak{C}(t)$, its loyalty score with respect to $\mathfrak{C}(t)$ is $S_{\mathfrak{C}(t)}(i) = \frac{|\{(i,j)|(i,j)\in\mathcal{E}_t\}|}{|\{(i,j)|(i,j)\in E\}|}$. The lower a node's loyalty score, the higher its reach outside of its cohort.

Why Community-based Defense? The *reachability* of a node is first and foremost defined by its local community neighborhood, as discussed in prior studies in contagion processes [12, 13, 24], which covers the k-hop neighborhood of a target node t for $k = 1, 2, ..., d(t, u)$ with $d(t, u)$ representing the eccentricity of t. For large k, the community's average loyalty score decreases and loses relevance to t. We argue that this is the case in the global target defense algorithms in which the target node's community structure is ignored- i.e., consideration of global neighborhoods instead of local neighborhood. The local defense strategies, on the other hand, can be considered another special case of community-based defense in which $k = 1$. However, social networks are shown to have high clusterability in their 2 and 3-hop neighborhoods as well [11, 13], and this is what a community-based method exploits. This gives a balance between capturing a larger search space, without the explosion in computation costs.

4.1 CoVerD Algorithm

CoVerD consists of four separate blocks. The **first block** (Algorithm 3) maximizes the loyalty score of the target node t by removing its connection to neighbors outside of its cohort in order of those neighbors' degrees. This ordering assures that even for a very limited budget, the target node loses its centrality effectively. The **second block** (Algorithm 1) is the degree-biased ROAM method [25] that iteratively disconnects the target from its highly connected neighbors. To assure the connectivity, as with the original ROAM algorithm, we make an edge between the disconnected neighbor and one of the immediate neighbors of t. The combination of these two blocks guarantees the high performance of the local perturbations in the absence of enough defense budget.

As the budget increases, the third and fourth block boost the performance of the algorithm and break the plateau of the local methods such as ROAM. The **third block** (Algorithm 4) increases the loyalty score of the $2-hop$ neighborhood of t by removing the $2-hop$ connections that leave $\mathfrak{C}(t)$. Increasing the loyalty score of this neighborhood promises to have a large impact on decreasing the closeness centrality of the target node (as shown in Figure 2). Although the third block achieves a considerable boost compared to using the first two blocks alone, we witnessed that, by building a *loyalty chamber* in the target's cohort, we are able to boost the performance of the algorithm for networks with densely connected communities (i.e., a low average loyalty score per community). We build the loyalty chamber in the **fourth block** (Algorithm 5) by using the remaining budget for disconnecting nodes within $\mathfrak{C}(t)$ whose difference in loyalty score is maximum. This last step divides the cohort

Algorithm 4: `Increase1HopLoyalty`

```
Input: $G$, $N$, $t$, $\mathfrak{C}(t)$, $b_d$
$flag \leftarrow True$; $spent \leftarrow 0$
for $p \in N$ do
    $N' \leftarrow$ SortByDegree($\mathcal{N}_G(p) \setminus \mathcal{N}_{\mathcal{G}_t}(p)$)
    for $q \in N'$ do
        $G' \leftarrow G(V, E \setminus \{(p,q)\})$
        if IsConnected($G'$) then
            $G \leftarrow G'$; $spent \leftarrow spent + 1$
            if $spent > b_d$ then
                break
            end
        end
    end
    if $spent > b_d$ then
        break
    end
    for $q \in \mathfrak{C}(t)$ do
        Compute $S_{\mathfrak{C}(t)}(q)$
    end
    for $q \in \mathcal{N}_{\mathcal{G}_t}(p)$ do
        if $S_{\mathfrak{C}(t)}(q) < 1$ then
            $G' \leftarrow G(V, E \setminus \{(p,q)\})$
            if IsConnected($G'$) then
                $G \leftarrow G'$; $spent \leftarrow spent + 1$
                if $spent > b_d$ then
                    break
                end
            end
        end
    end
    if $spent > b_d$ then
        break
    end
end
$b_d \leftarrow b_d - spent$
if $b_d \leq 0$ then
    $flag \leftarrow False$
end
return $G, b_d, flag$
```

Algorithm 5: `BuildLoyaltyChamber`

```
Input: $G$, $t$, $\mathfrak{C}(t)$, $b_d$
$flag \leftarrow True$; $spent \leftarrow 0$
for $p \in \mathfrak{C}(t)$ do
    compute $S_{\mathfrak{C}(t)}(p)$
end
$candid \leftarrow \{(n_1, n_2) | (n_1, n_2) \in \mathcal{E}_t, S_{\mathfrak{C}(t)}(n_1) = 1, S_{\mathfrak{C}(t)}(n_2) < 1\}$
$candid \leftarrow$SortByScoreDiff($candid$)
for $q \in candid$ do
    $G' \leftarrow G(V, E \setminus \{(p,q)\})$
    if IsConnected($G'$) then
        $G \leftarrow G'$ ; $spent \leftarrow spent + 1$
        if $spent > b_d$ then
            break
        end
    end
end
$b_d \leftarrow b_d - spent$
if $b_d \leq 0$ then
    $flag \leftarrow False$
end
return $G, b_d, flag$
```

into two distinctive partitions without disconnecting the graph; the loyal nodes that are purely connected between themselves, and the disloyal nodes that are loosely attached to the cohort. The overall algorithm is shown in Algorithm 2.

Time Complexity. For a target node t, Algorithms 3 and 1 visit at most $|\mathcal{N}_G(t)|$ nodes each. The Algorithm 4 iterates over the 2-hop neighborhood of the target and computes the loyalty score for each node in the cohort, visiting on average $|\mathcal{N}_G(t)| \cdot (\overline{|\mathcal{N}_G|} + |\mathfrak{C}_t|)$, in which the $\overline{|\mathcal{N}_G|}$ is the average degree in G. The Algorithm 5 visits every node in the cohort once to re-compute their loyalty score and visits exactly $|\mathfrak{C}_t|$ nodes. For the average case in which $|\mathcal{N}_G(t)| \approx \overline{|\mathcal{N}_G|} = \mathfrak{d}$, the overall time complexity of *CoVerD* is $\mathcal{O}((\mathfrak{d}+1) \cdot |\mathfrak{C}_t| + \mathfrak{d}^2 + 2\mathfrak{d})$, which for $\mathfrak{d} \ll |\mathfrak{C}_t|$ curtails to $\mathcal{O}(\mathfrak{d} \cdot |\mathfrak{C}_t|)$. So, the speed of *CoVerD* depends mainly on the size of the target's community. This is a significant improvement from the polynomial time complexity of global methods (see Section 2) and is still comparable to local methods with average time complexity of $\mathcal{O}(\mathfrak{d})$.

5 Experiments

In this section, we analyze the performance of *CoVerD* algorithm against both local and global perturbation algorithms on five real-world datasets.

Experimental Setup. In our experiments, we implement a defense strategy (edge perturbations) on a given network G for a target node t to obtain *defended* network G^*. Then, we run a crawling algorithm starting from a given source node and obtain b_a, the number of nodes explored by the adversary crawler, before reaching t. b_a is at most $|V|$, so the defender performance metric is $\frac{b_a}{|V|}$. We select target nodes in three ways, and for each, select 5 nodes:

- **Random Targets:** The target nodes are chosen uniformly at random.
- **Degree-based Targets:** The targets are chosen with probability proportional to degree. This strategy mirrors the attack on well-connected influential nodes.
- **Community-based Targets:** First, each community receives two scores, each in $[0, 1]$, based on (a) their size and (b) density of their intra-group edges. The normalized sum of these two scores gives a final ranking of each community. The targets are chosen from $|V|$ with a probability that is biased towards the score of their respective community. This strategy mirrors the attack on well-connected influential communities.

We choose five different source nodes at random for the attacker's starting point. We run our simulation for all (target, source) combinations (i.e., 75 pairs) and report their average performance. This procedure is repeated for values of b_d ranging from 0.1% to 5% of the edges in the graph.

Datasets. We use five real-world datasets whose names and basic statistics are shown in Table 1. For the community assignment in each network, we use Louvain community detection method [7].

Defender algorithms. We compare our algorithm against both local and global defenders. ROAM is the most prominent local-perturbation method [25]

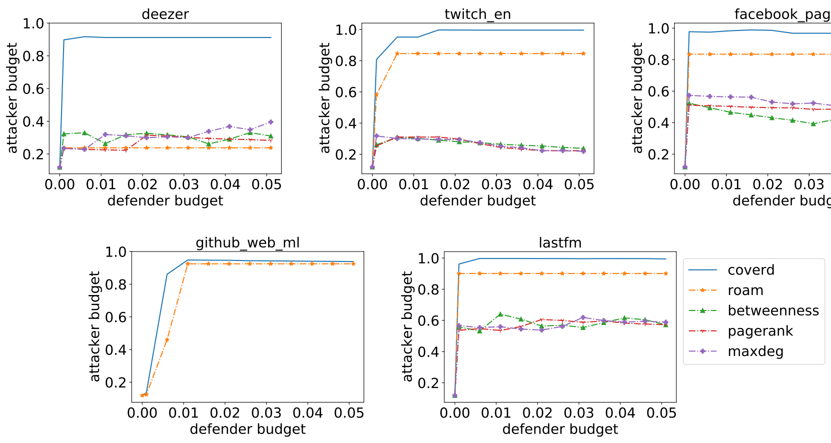

Fig. 1: *The defender budget vs. attacker budget for different defender algorithms. The plots show the aggregated simulation results for degree-based target nodes and BFS crawling attack. Similar results were obtained for DFS attack as well as community-based and random target nodes. CoVerD outperforms all the baselines for the same values of b_d. It also reaches the optimal performance ($b_a \approx 1$) on the majority of datasets.*

(see 2). Among the global perturbations, however, our choices are limited to those that have feasible computation time on our real-world networks. We build three global defenders by following the proposed approximate global perturbation methods in [3, 16]. In our four baselines: **ROAM** is implemented similarly to Algorithm 1, except that we do not limit the neighborhood of the target to the cohort neighbors. **Betweenness, PageRank**, and **MaxDegree** score each edge as the sum of it's endpoints' scores, where each node is scored by its betweenness centrality, PageRank, or MaxDegree, respectively. The top scoring b_d edges are removed.

Attacker algorithms. According to [17], the hallmark of aggressive crawling is the choice of an expansion-based method that allows for as far as possible from the starting point, such as depth-first search (DFS). On the other hand, innocent crawlers tend to remain in the local neighborhood of the starting node, and tend to resemble breadth-first search (BFS). As such, we have chosen these two crawling techniques to show the performance of our algorithm in the presence of both aggressive and innocent crawlers. (Note, however, that *CoVerD* is agnostic to the crawling algorithm.)

5.1 Results

Table 1 shows the result[3] of our experiments for $b_d = 0.006|V|$. Against the BFS crawler, *CoVerD* shows a pronounced superior performance and, in some instances, it increases the crawler's budget by 3 and 10 times compared to the

[3] For our largest dataset, `github`, obtaining the results of the global measures was infeasible

Networks	**Defender**	**Random Target**		**Degree-based Target**		**Community-based Target**	
		BFS	DFS	BFS	DFS	BFS	DFS
lastfm-asia $\lvert V\rvert = 7,624$ $\lvert E\rvert = 27,806$	Original	0.12	0.28	0.54	0.20	0.48	0.21
	Betweenness	0.16	0.38	0.53	0.27	0.49	0.37
	PageRank	0.23	0.23	0.55	0.19	0.5	0.34
	MaxDegree	0.18	0.38	0.55	0.27	0.43	0.33
	ROAM	0.44	0.55	0.90	**0.58**	0.84	0.40
	CoVerD	**0.99**	**0.68**	**1.00**	0.56	**0.96**	**0.50**
musae-twitch $\lvert V\rvert = 7,126$ $\lvert E\rvert = 35,324$	Original	0.19	0.86	0.35	0.18	0.53	0.43
	Betweenness	0.20	0.88	0.31	0.17	0.55	**0.69**
	PageRank	0.26	0.88	0.31	0.20	0.55	0.47
	MaxDegree	0.31	0.69	0.30	0.18	0.54	0.37
	ROAM	0.45	0.86	0.84	0.29	**0.89**	0.58
	CoVerD	**0.48**	**0.94**	**0.95**	**0.78**	**0.89**	0.53
deezer-europe $\lvert V\rvert = 28,281$ $\lvert E\rvert = 92,752$	Original	0.20	0.26	0.24	0.57	0.12	0.22
	Betweenness	0.23	0.18	0.33	0.67	0.11	0.16
	PageRank	0.19	0.18	0.23	0.54	0.11	0.23
	MaxDegree	0.19	0.16	0.23	0.39	0.12	0.15
	ROAM	0.56	0.52	0.23	0.39	0.42	0.52
	CoVerD	**0.82**	**0.56**	**0.92**	**0.93**	**0.99**	**0.79**
musae-facebook $\lvert V\rvert = 22,470$ $\lvert E\rvert = 171,002$	Original	0.32	0.38	0.49	0.39	0.8	0.69
	Betweenness	0.33	0.36	0.49	0.37	0.76	0.68
	PageRank	0.31	0.51	0.51	0.33	0.80	0.70
	MaxDegree	0.33	0.17	0.56	0.40	0.79	0.59
	ROAM	0.85	**0.71**	0.83	**0.65**	0.80	0.58
	CoVerD	**0.99**	0.65	**0.98**	**0.65**	**0.98**	**0.89**
musae-github $\lvert V\rvert = 37,700$ $\lvert E\rvert = 289,003$	Original	0.27	0.35	0.00	0.01	0.33	0.18
	Betweenness	NA	NA	NA	NA	NA	NA
	PageRank	NA	NA	NA	NA	NA	NA
	MaxDegree	NA	NA	NA	NA	NA	NA
	ROAM	0.93	0.52	0.46	0.05	0.90	0.49
	CoVerD	**0.98**	**0.64**	**0.86**	**0.59**	**0.97**	**0.53**

Table 1: *The performance of all defenders against BFS and DFS crawling attacks for different types of target nodes. The values show the normalized attacker budget ($\frac{b_a}{|V|}$) in order to discover the target node. The values closer to* 1 *indicate superior performance of the defender and are shown in* ***bold****. For BFS crawlers, CoVerD always surpasses the benchmarks with considerable Margie. The same holds true for DFS crawlers in the majority of cases. In general, all defenders perform worse against the DFS crawling attack (aggressive crawling).*

next best-performing benchmark (see the result for degree-based target node of `deezer` and `github`). In 60% of simulations, it achieves close to perfect results ($b_a \geq 0.96$). The hardest dataset for this task was `twtich` in which the results of our best performing models, *CoVerD* and ROAM, are relatively low. However, compared to the undefended graph and the three global methods, **CoVerD** still increases the attacker's budget by $\approx 50\%$. Against the more aggressive crawling scheme of DFS, all models perform worse than their BFS counterpart, as expected. Nonetheless, *CoVerD* outperforms all benchmarks in 80% of simulations and, in the remaining cases, it offers competitive results.

Figure 1 shows the change in defenders' performance with respect to their availbe budget. In all cases, *CoVerD* reaches near optimal performance with budgets less than $0.01|V|$. ROAM also offers decent results in the majority of the cases. However, its effectiveness rapidly reaches its plateau and never offers a near-optimal result. Its surprisingly poor performance for `deezer` in Figure 1, in contrast to the near-optimal performance of **CoverD**, suggests the importance of looking beyond the immediate neighborhood of the target node. Recall that for small defense budgets, the only difference between *CoVerD* and ROAM is

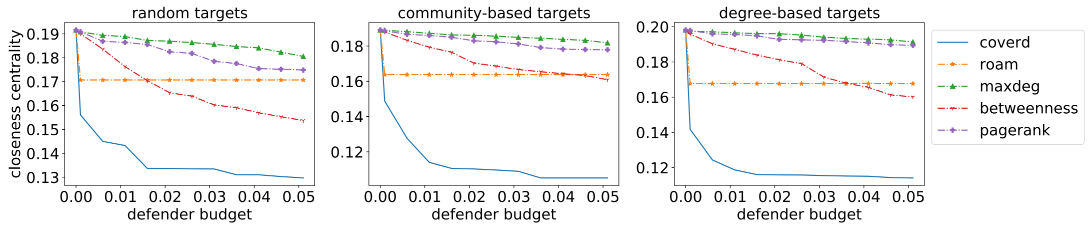

Fig. 2: *Closeness centrality of the target node (y-axis) vs. the defense budget (x-axis). The plots belong to* `lastfm` *data. For each plot, we have used the mean of the closeness centrality among all the target nodes. It is evident that CoVerD substantially surpasses both local and global measures in decreasing the closeness centrality of the targets for all target types.*

the maximization of target's loyalty score, $S_{\mathfrak{C}(t)}(t)$. Hence, the superior performance of *CoVerD* for small budgets versus that of ROAM in Figure 1 shows the importance of community membership in determining a node's reachability.

In all previous studies, the indicator of a defender's success was defined by its ability to minimize the closeness (or betweenness) centrality of a target node (in contrast to ours in which the increase in b_a marks the performance). We also show the change in the closeness centrality of the target nodes for different b_d in figure 2. Even though we did not use any global measures to decrease the closeness centrality directly, *CoVerD* has achieved the fastest and deepest drop in the target's centrality by focusing only on its local community structure. This figure also shows that for achieving comparable performance with [already computationally expensive] global defenders, such as the Betweenness model, we need to invest in larger defense budgets (note the slow but steady decrease of the centrality for Betweenness model in Figure 2).

6 Conclusion & Future Direction

In this study, we formalized the problem of vertex protection from a defender's perspective. We proposed the *CoVerD* heuristic that leverages the community structure of social networks. This algorithm retains the fast computation of local network perturbations and shows superior performance compared to both local and global defenders. Despite using only the local community information, our algorithm achieves a substantially lower closeness centrality than both local and global perturbation models.

Future Direction. This study is an important step forward in the field of network protection and privacy to focus on heuristics that are both practical and efficient in the real-world settings. Two valuable extensions to *CoVerD* are (1) investigating the correlation between different community structures and defender's performance; (2) introducing additional constraints to the defender's decision making, such as maintaining certain properties of the network or avoiding the formation of certain motif subgraphs.

Bibliography

1. Abrahamsson, O.: Hide and Seek in a Social Network. Master's thesis, Linköping University, Sweden (2017)
2. Areekijseree, K., Laishram, R., Soundarajan, S.: Max-node sampling: An expansion-densification algorithm for data collection. In: IEEE International Conference on Big Data (2016)
3. Areekijseree, K., Laishram, R., Soundarajan, S.: Guidelines for online network crawling: A study of data collection approaches and network properties. In: Proceedings of the 10th ACM Conference on Web Science (2018)
4. Avram, M.V., Mishra, S., Parulian, N.N., Diesner, J.: Adversarial perturbations to manipulate the perception of power and influence in networks. In: 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). IEEE (2019)
5. Bai, Q., Xiong, G., Zhao, Y., He, L.: Analysis and detection of bogus behavior in web crawler measurement. Procedia Computer Science 31 (2014)
6. Bisenius, P., Bergamin, E., Angriman, E., Meyerhenke, H.: Computing top-k closeness centrality in fully-dynamic graphs. In: 2018 Proceedings of the Twentieth Workshop on Algorithm Engineering and Experiments (ALENEX). SIAM (2018)
7. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment 2008(10), P10008 (2008)
8. Borassi, M., Crescenzi, P., Marino, A.: Fast and simple computation of top-k closeness centralities. arXiv preprint arXiv:1507.01490 (2015)
9. Crescenzi, P., d'Angelo, G., Severini, L., Velaj, Y.: Greedily improving our own centrality in a network. In: International Symposium on Experimental Algorithms. Springer (2015)
10. Dougnon, R.Y., Fournier-Viger, P., Nkambou, R.: Inferring user profiles in online social networks using a partial social graph. In: Canadian Conference on Artificial Intelligence (????)
11. Fortunato, S.: Community detection in graphs. Physics reports 486(3-5) (2010)
12. Gupta, N., Singh, A., Cherifi, H.: Community-based immunization strategies for epidemic control. In: 2015 7th international conference on communication systems and networks (COMSNETS). IEEE (2015)
13. Hozhabrierdi, P., Zhu, R., Onyewu, M., Soundarajan, S.: Network-based analysis of early pandemic mitigation strategies: Solutions, and future directions. Northeast Journal of Complex Systems (NEJCS) 3(1) (2021)
14. Ji, J., Wu, G., Duan, C., Ren, Y., Wang, Z.: Greedily remove k links to hide important individuals in social network. In: International Symposium on Security and Privacy in Social Networks and Big Data. Springer (2019)
15. Kumar, M., Bhatia, R., Rattan, D.: A survey of web crawlers for information retrieval. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 7(6) (2017)

16. Laishram, R., Hozhabrierdi, P., Wendt, J., Soundarajan, S.: Netprotect: Network perturbations to protect nodes against entry-point attack. In: 13th ACM Web Science Conference 2021 (2021)
17. Mondal, M., Viswanath, B., Clement, A., Druschel, P., Gummadi, K.P., Mislove, A., Post, A.: Defending against large-scale crawls in online social networks. In: Proceedings of the 8th international conference on Emerging networking experiments and technologies (2012)
18. Narayanan, A., Shmatikov, V.: De-anonymizing social networks. In: IEEE symposium on security and privacy. IEEE (2009)
19. Okamoto, K., Chen, W., Li, X.Y.: Ranking of closeness centrality for large-scale social networks. In: International workshop on frontiers in algorithmics. Springer (2008)
20. Olsen, P.W., Labouseur, A.G., Hwang, J.H.: Efficient top-k closeness centrality search. In: 2014 IEEE 30th International Conference on Data Engineering. IEEE (2014)
21. Rüdian, S., Pinkwart, N., Liu, Z.: I know who you are: Deanonymization using facebook likes. In: Workshops der INFORMATIK 2018-Architekturen, Prozesse, Sicherheit und Nachhaltigkeit. Köllen Druck+ Verlag GmbH (2018)
22. Schlicher, B.G., MacIntyre, L.P., Abercrombie, R.K.: Towards reducing the data exfiltration surface for the insider threat. In: 2016 49th Hawaii International Conference on System Sciences (HICSS). IEEE (2016)
23. Shang, Y.: False positive and false negative effects on network attacks. Journal of Statistical Physics 170(1) (2018)
24. Wang, M., Tan, Q., Wang, X., Shi, J.: De-anonymizing social networks user via profile similarity. In: 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC). IEEE (2018)
25. Waniek, M., Michalak, T.P., Wooldridge, M.J., Rahwan, T.: Hiding individuals and communities in a social network. Nature Human Behaviour 2(2) (2018)