2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

HackerScope: The Dynamics of a Massive Hacker Online Ecosystem

Risul Islam
UC Riverside
risla002@ucr.edu

Md Omar Faruk Rokon
UC Riverside
mroko001@ucr.edu

Ahmad Darki
UC Riverside
adark001@ucr.edu

Michalis Faloutsos
UC Riverside
michalis@cs.ucr.edu

***Abstract*—Authors of malicious software are not hiding as much as one would assume: they have a visible online footprint. Apart from online forums, this footprint appears in software development platforms, where authors create publicly-accessible malware repositories to share and collaborate. With the exception of a few recent efforts, the existence and the dynamics of this community has received surprisingly limited attention. The goal of our work is to analyze this ecosystem of hackers in order to: (a) understand their collaborative patterns, and (b) identify and profile its most influential authors. We develop HackerScope, a systematic approach for analyzing the dynamics of this hacker ecosystem. Leveraging our targeted data collection, we conduct an extensive study of 7389 authors of malware repositories on GitHub, which we combine with their activity on four security forums. From a modeling point of view, we study the ecosystem using three network representations: (a) the author-author network, (b) the author-repository network, and (c) cross-platform egonets. Our analysis leads to the following key observations: (a) the ecosystem is growing at an accelerating rate as the number of new malware authors per year triples every 2 years, (b) it is highly collaborative, more so than the rest of GitHub authors, and (c) it includes influential and professional hackers. We find 30 authors maintain an online "brand" across GitHub and our security forums. Our study is a significant step towards using public online information for understanding the malicious hacker community.**

***Index Terms*—GitHub, Hackers, Community, Egonet**

I. Introduction

"How can a 17 year old kid from Florida [1] be reportedly the mastermind behind the recent hacking of Twitter? This question is part of the motivation behind this work.

The security community has a fairly limited understanding of malicious hackers and their interactions. As a result, security practitioners do not really know their "enemy". On the one hand, the hacker community is fairly wide encompassing curious teenagers, aspiring hackers, and professional criminals. On the other hand, the hackers are surprisingly bold in leaving a digital footprint, if one looks at the right places in the Internet. For example, there are various online forums, where hackers not only share information, but they also boast of their successes.

How can we begin to understand the ecosystem of malicious hackers based on their online footprint? The input is the online activities of these hackers, and the goal is to answer the following questions: (a) do these hackers work in groups or alone, and (b) who are the most influential hackers? Here, we consider two types of platforms that hackers frequent: (a) software archives, and (b) online security forums. It turns out that popular and public software archives, such as GitHub harbor **malware authors**, who create publicly-accessible malware repositories [2]. Furthermore, online forums have recently emerged as marketplaces and information hubs of malicious activities [3], [4]. In the rest of this paper, we will use the term **hacker** to refer to actors who develop and use software of malicious intent. We will also use the term *hackers* and *malware authors* interchangeably, although some malware authors may not have malicious intent.

There is limited work for the problem as defined above. First, we are not aware of a study that systematically profiles the dynamics of the online hacker ecosystem, and especially one considering software archives. Most of the previous efforts on GitHub follow a software-centric view or study GitHub at large without focusing on malware [5] [6] [7]. Most of the previous works on online forums focus on identifying emerging topics and threats [3], [4]. Other efforts report malware activity, focusing on hacking events, and much less, if at all, on the ecosystem of hackers [8], [9]. We elaborate on previous works in Section VIII.

We propose **HackerScope**, a systematic approach for modeling the ecosystem of malware authors by analyzing their online footprint. We start with an extensive analysis of malware authors on GitHub, as this is a significantly less-studied space. We then use security forums to find more information about these authors. From an algorithmic point of view, we use three network representations: (a) the author-author network, (b) the author-repository network, and (c) cross-platform egonets, which we explain later. In addition, we use some basic Natural Language Processing techniques, which we intend to develop further in the future.

We apply and evaluate our approach using 7389 malware authors on GitHub over the span of 11 years and leverage the activity on four security forums in the grey area between white-hat and black-hat security. GitHub is arguably the largest repository with roughly 30 million public repositories, while, appropriately fine-tuned, our approach can be used on other software archives. Our approach encompasses four research thrusts, which identify and model: (a) statistics and trends, (b) communities of hackers and their dynamics, (c) influential hackers, and (d) hacker profiles across different online platforms. For the latter type, we show the collaborators of hackers as captured by the cross-platform egonets spanning GitHub and security forums in Figure 1. Our key results are summarized in the following points.

a. The ecosystem is growing at an accelerating rate: The

IEEE/ACM ASONAM 2020, December 7-10, 2020
978-1-7281-1056-1/20/$31.00

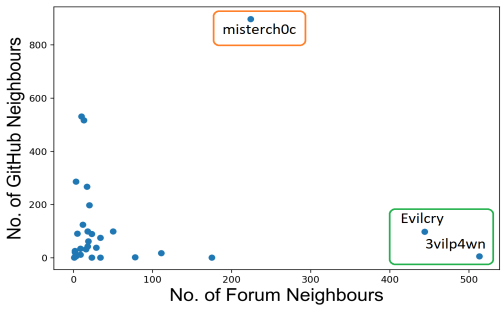

Figure 1: Profiling hackers across platforms using our cross-platform egonet: the scatter-plot of the number of neighbors on GitHub versus those on security forums for 30 malware authors as captured in our cross-platform egonet.

number of new malware authors on GitHub is roughly tripling every two years. This alarming trend points to the importance of monitoring this ecosystem.

b. The ecosystem is highly collaborative: We find 513 collaboration communities on GitHub with high cohesiveness (*Modularity Score* within [0.65-0.78]), including many large communities with hundreds of users. The malware community is very collaborative: a malware repository is forked *four times* more compared to a regular GitHub repository.

c. We identify a group of 1.7% of influential authors: We develop a systematic approach to determine the influence among malware authors. Our novelty lies in: (a) considering many types of interactions, and (b) capturing the network-wide influence of an author. We find a core group of 1.7% of the malware authors, who are responsible for: (a) generating influential repositories, and (b) providing the social backbone of the malware community.

d. We identify professional hackers in the ecosystem: We find that 30 authors are professional *malicious* hackers. Going across platforms, we find GitHub authors who are quite active on our security forums. We show the evidence that these are professional hackers, who are building an online "brand". For example, user *3vilp4wn* is the author of a keylogger repository on GitHub, which he promotes in the *HackThisSite* forum using the same username (shown at bottom right in Figure 1).

Our work in perspective. The proposed work is part of an ambitious goal: we want to model the Internet hacker ecosystem at large as it manifests itself across platforms. Our initial results are promising: a) the hackers seem to want to establish a brand, hence they want to be visible, and b) a cross-platform study is possible, as some authors maintain the same login name. Our systematic approach here constitutes a building block towards the ultimate goal. With appropriate follow up work, achieving this goal can have a huge practical impact: security analysts could prepare for emerging threats, anticipate malicious activity, and identify their perpetrators.

Open-sourcing for maximal impact. We use Python v3.6.2 packages to implement all the modules of HackerScope. We intend to make our datasets and tools public for research purposes.

II. Background and Data

Our work focuses on GitHub, the largest software archive with roughly 30 million public repositories, and uses data from security forums. Although GitHub policies do not allow malware, authors do not seem to abide by them.

A. GitHub data. GitHub platform enables software developers to create software repositories in order to store, share, and collaborate on projects and provides many social-network-type functions.

We define some basic terminology here. We use the term *author* to describe a GitHub user who has created at least one repository. A *malware repository* contains malicious software and a *malware author* owns at least one such repository. Users can *star, watch* and *fork* other *malware repositories*. *Forking* means creating a clone of another repository. A forked repository is sometimes merged back with the original parent repository, and we call this a *contribution*. Users can also *comment* by providing suggestions and feedback to other authors' repositories.

We use a dataset of 7389 malware authors and their related 8644 malware repositories, which were identified among 97K repositories in our prior work [2]. This is arguably the largest malware archive of its kind with repositories spanning roughly 11 years. These repositories have been identified as malicious with a very high precision (89%). Note that the queries with the GitHub API, which were used in the data collection, return primary or non-forked repositories. A discussion on the process, accuracy, and validity of the dataset can be found in the original study [2].

For each malware author in our dataset, we have the following information: (a) the list of the malware repositories created by her, and (b) the list of followers. For each malware repository, we have the lists of users, who: (a) star, (b) watch, (c) fork, (d) comment, or (e) contribute to the repository.

Repository metadata. Each repository is also associated with a set of user generated fields, such as title, readme file, description. We can use this *metadata* to extract information about the repository. We leverage our earlier work where we discuss the processing of this metadata in more detail [2].

For a given repository, a security expert would want to know: (a) the type of malware (e.g. ransomware and keylogger), and (b) the target platform (e.g. Linux and Windows). For this, we define two sets of keywords: (a) 13 types of malware, S_1 and (b) 6 types of target platforms, and S_2. Figure 6 provides a visual list of these two sets of keywords. We define the Repository Keyword Set, W_r, for repository r, as a set consisting of the keyword sets S_1 and S_2 that are present in its metadata. Clearly, one can extend and refine these keyword sets, to provide additional information, such as the programming language in use, which we will consider in the future. Note that our earlier work provides evidence that using this metadata as we do here can provide fairly accurate and useful information [2].

B. Security forum data. We also utilize data that we collect from four security forums: Wilders Security, Offensive Community, Hack This Site, and Ethical Hackers [10]. In

Table I: Our four online security forums.

Forum	Users	Threads	Posts
Offensive Comm.	5412	3214	23918
Ethical Hacker	5482	3290	22434
Hack This Site	2970	2740	20116
Wilders Security	3343	3741	15121

these forums, users initiate discussion threads in which other interested users can post to share their opinion. Each tuple in our dataset contains the following information: forum ID, thread ID, post ID, username, and post content. We provide a brief description of our forums below, and an overview of key numbers in Table I.

a. OffensiveCommunity (OC): As the name suggests, this forum contains "offensive security" related threads, namely, breaking into systems. Many posts consist of step by step instructions on how to compromise systems, and advertise hacking tools and services.

b. HackThisSite (HTS): As the name suggests, this forum has also an attacking orientation. There are threads that explain how to break into websites and systems, but there are also more general discussions on cyber-security.

c. EthicalHackers (EH): This forum seems to consist mostly of "white-hat" hackers, as its name suggests. However, there are many threads with malicious intentions in this forum.

d. WildersSecurity (WS): The threads in this forum fall in the grey area, discussing both "black-hat" and "white-hat" skills.

III. Our Approach

We have an ambitious vision for our approach, which we plan to release as a software platform. We provide a brief overview in Figure 2. In this paper, we will elaborate on the four analysis modules: (a) a statistics and trends module, which provides the landscape of primary behaviors of the ecosystem (Section IV), (b) a community analysis module, which identifies and profiles communities of collaboration (Section VI), (c) an influence analysis module, which defines and calculates the significance of authors (Section V), and (d) cross-platform analysis module (Section VII).

In addition, our approach also includes: a data collection module, which aggregates, cleans and preprocesses the raw information; a control center module; and a reporting module. These modules are not equally developed, while at the same time, we could not provide all the types of results that we have available due to space limitations.

Below, we highlight some interesting or novel aspects of our approach, which are often cutting across several modules.

a. Synthesizing multi-source data. Our approach focuses on data for authors from GitHub and combines it with additional data from security forums, and Internet searches.

b. Defining appropriate features. As we already saw, the authors and the repositories have a very rich set of interactions. We have primary (measured directly) and secondary (derived from the primary) features, which need to be determined carefully to capture effectively the dynamics of the ecosystem. These interactions go beyond a simple "friend" relationship of other social media.

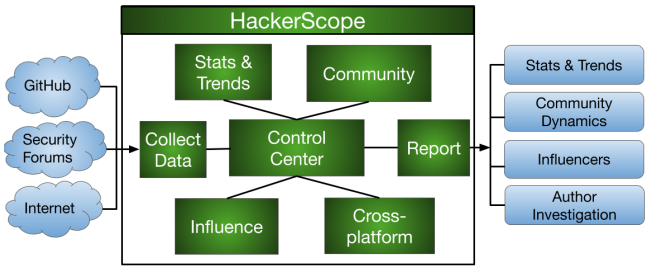

Figure 2: The overview of our approach highlighting the key functions.

c. Modeling the dynamics. We use three network representations to capture the rich interactions and relationships among authors and repositories. The network representations include: (a) the author-author network, (b) the author-repository network, and (c) cross-platform egonets.

d. Reporting behaviors. The goal is to provide intuitive and actionable information in an appealing and ideally interactive fashion. The results in this paper provide an indication of some initial plots and tables that our approach will provide to the end user, who could be a researcher or a security analyst.

IV. Statistics and Trends

This section describes the functionality of the *statistics and trends* module of our approach, whose intention is to provide a basic understanding of author behaviors.

A. Basic distributions of malware authors. We study the complementary cumulative distribution function (CCDF) of three metrics: (a) the number of repositories created, (b) the number of followers, and (c) sum of the number of forks across all the malware repositories of the author. As expected all distributions are skewed, but the plots are omitted due to space constraints. First, we find that 15 authors are contributing roughly 5% of all malware repositories, while 99% of all authors have created less than 5 repositories each. Second, we find that 3% (221) of the authors have more than 300 followers each, while 70% of the authors have less than 16 followers. Finally, examining the total number of forks per author, we find that 3% (221) of the authors have their repositories forked more than 150, while 43% of authors encounter at least one fork.

B. Forking behavior: Malware repositories are forked four times more than the average repository. Malware repositories are more aggressively forked, which is an indication of the higher collaboration in the ecosystem. First, we find that a malware repository is forked 4.01 times on average, while a regular GitHub repository is forked 0.9 times, as reported in previous studies [11]. Second, we want to see if this is due to a few popular repositories, but this is not the case. We find that 39% of the malware repositories are forked at least once, while this is true for only 14% for general repositories [11].

C. Trends. *"How fast is this ecosystem growing?"* To answer the question, we plot the number of new malware authors per year in Figure 3. We consider that an author joins the ecosystem at the time that they create their first malware

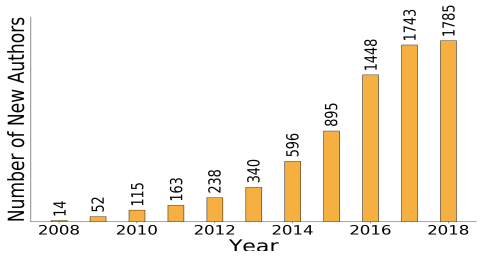

Figure 3: New malware authors in the ecosystem per year.

repository in our database.

a. The number of new malware authors almost triples every two years. We plot the new malware authors per year in Figure 3. We observe an increase from 238 malware authors in 2012 to 596 authors in 2014 and to 1448 authors in 2016. We also observe a steep 62% increase from 2015 to 2016. This trend is interesting and alarming at the same time.

b. The number of new malware repositories more than triples every four years. Echoing the growth of the authors, the number of repositories is also increasing super-linearly. In the future, we plan to study the trends of malware in terms of both types of malware and its target platform.

V. Identifying Influential Authors

To understand the dynamics of the ecosystem, we want to answer the following question: *"Who are the most influential authors?"* The functionality in this section is part of the *influence analysis* module of Figure 2.

A. HackerScore: Identifying influential authors. We argue that finding influential authors presents several challenges. First, there are many different activities and interactions, such as creating repositories, commenting, following other authors and being followed by other authors. Second, we can consider two types of actions: (a) creating influential artifacts, (b) observing and engaging with other people and artifacts. Furthermore, the distinction is not always clear. For example, forking a repository creates a new, but derivative, repository.

To address the above challenges, we take **socially-aware approach** to influence: creating a few influential repositories is more important than creating many non-influential repositories. We discuss how we model and calculate this influence below.

The Author-Author graph (AA). We create the Author-Author network to capture the network-wide interaction among authors. We define a weighted labeled multi-digraph: $G(V, E, W, L_e)$ where V is the malware author set, E is the set of edges, W is the weight set and L_e is the set of labels that an edge e can be associated with. These labels correspond to different types of relationships between authors. Here we opted to consider only malware authors in the graph to raise the bar for being part of the hacker community.

The types of interactions. We consider four types of relationships between authors here. A directed edge (u, v) from author u to v can be (i) a follower edge: when u follows v, (ii) a fork edge: when u forks a repository of v, (iii) a contribution edge: u contributes code in a repository of v, and (iv) a comment edge: u comments in a repository of v. These relationships capture the most substantial author-level interactions.

The multi-graph challenge and weight calibration. Our graph consists of different types of edges, which represent different relationships that we want to consider in tandem. The challenge is that the relationships have significantly different distributions, which can give an unfair advantage or eliminate the importance of a relationship. For example, contribution activities are rarer compared to following, but one can argue that a contribution to a repository is a more meaningful relationship and it should be given appropriate weight.

For fairness, we make the weight of a type of edge inversely proportional to a measure of its relative frequency. In detail, we calculate the average degree d_{type} for each *type* of edge: follower, fork, contribution, and comment from the subgraph containing only that type of edges from the AA graph. We find the following average degrees: $d_{follower} = 12.21$, $d_{fork} = 4.67$, $d_{contribution} = 0.53$ and $d_{comment} = 0.49$. We normalize these average degrees using the minimum average degree ($d_{min} = 0.49$) and we get the *inverse* of this value as the weight for that edge, namely, d_{min}/d_{type}. This way, we set the following weights: $w = 0.04$ for a following edge, $w = 0.1$ for a forking edge, and $w = 1$ for a commenting or a contribution edge. This enables us to consider each relationship type more fairly and meaningfully.

We propose a socially-aware and integrated approach to combine all the author activities in a single framework. First, we identify and define two roles in the ecosystem: (a) **producers**, who create influential malware repositories, and (b) **connectors**, who enhance the community by engaging with influential malware authors and repositories. To calculate the roles of the malware authors, we first model the interaction among authors in the AA graph described above. Next, we apply our algorithm, a customized version of a weighted hyperlink-induced topic search algorithm modified to handle the multiple types of relationships between authors. We discuss the related algorithms in Section VIII.

Calculating the HackerScore. We associate each node u with two values: (a) a **Producer HackerScore** value, PHS_u, and (b) **Connector HackerScore** value, CHS_u. Let $w(u, v)$ be the weight of edge (u, v) based on its label, as discussed above.

The algorithm iterative refines the producer and connector values until it converges. We, now, elaborate on the steps. First, PHS_u and CHS_u are initialized to 1. During the iterative step, the algorithm updates the values as follows: (i) for all v pointing to u: $PHS_u = \sum_v w(v, u) * CHS_v$, or zero in the absence of such edges, (ii) for all z pointed by u: $CHS_u = \sum_z w(u, z) * PHS_z$, or zero in the absence of such edges, and (iii) we normalize PHS_u and CHS_u, so that $\sum_u PHS_u = \sum_u CHS_u = 1$. For the convergence, we set a tolerance threshold of 10^{-9} for the change of the value of any node. After 449 iterations, we obtain the two HackerScore values for each author.

Identifying influential malware authors. In Figure 4, we plot the Connector HackerScore versus Producer HackerScore for our malware authors. Separately, we identify "knees" in the

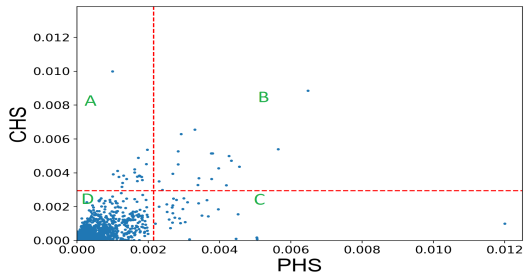

Figure 4: The scatterplot of the Connector HackerScore vs. Producer HackerScore for the malware authors in our GitHub dataset.

individual distributions of each score at PHS = 0.00215 and CHS = 0.0029 indicated by the red dotted lines. This way, we observe four regions defined by the combination of low and high values for PHS and CHS values which shows if an author is influential as producer or connector.

A few authors (1.7%) drive the community. The three regions of influence together consist of 128 malware authors (1.7%). The break down of the region size is fairly even: Region A of mostly connector authors devoted to connect the malware community is 0.6%, Region C of the influential producers who are the originator of the malware resources is 0.7%, and Region B of dual influence is 0.4%. We use the term **Highly Influential Group (HIG)** to refer to this group of authors.

We provide a profile overview of the two most influential authors per region in Table II. The most influential author of Region C is *cyberthrets*, with the highest PHS (0.012) and 336 malware repositories. She gained a huge following by creating all her repositories of assembly code malware on Feb 16, 2016. The top connector author from Region A is *critics* with a CHS score of 0.01, which stems from her 446 comments across 301 repositories. The top malware author from Region B is *D4Vince* for his dual role in producing credential reuse tools with 7 repositories and 165 comments and 187 contributions.

The importance of socially-aware significance. We argue that our socially-aware definition of significance provides more meaningful results than simply taking the top-ranked users in any primary metric in isolation. First, the two scores capture different aspects of influence: they can differ by orders of magnitude as is the case with *cyberthrets* and *ytisf*. Second, our scores capture a combined network-wide influence that each primary metric could miss. For example, our most influential producers do not always own many malware repositories. Malware author *D4vince* and *n1nj4sec*, mentioned in Table II, have single-digit repositories (7 and 8 respectively) and yet are two of the top producers. On the other hand, author *kaist-is521* is ranked way below than *n1nj4sec* in terms of HackerScore (PHS =0.0001 and CHS =0.00013), although she has 18 malware repositories.

B. Reciprocity of interactions. We want to understand better the nature of the author interactions here.

"Is the influence among malware authors reciprocal?" The answer is negative: **the relationships are not reciprocal**, which is in stark contrast to the reciprocal relationships in other social media like Twitter and Facebook [12]. We consider a total of six relationships: following, forking, commenting, contributing, watching, and starring relationships. We define the **Reciprocity Index** for relationship x, RI_x, to be the ratio of reciprocal relationships over the pairs of authors with that type of relationship (unilateral or mutual) in the Author-Author network.

Table II: The profiles of the two most influential malware authors from each region A, B, and C.

Name	PHS	CHS	Repos	Follow-ers	Forks	Com-ments	Cont-rib/s
cyberthrets	**0.012**	0.001	336	1013	778	13	2
ytisf	**0.005**	10^{-6}	12	606	1412	4	1
critics	0.001	**0.01**	6	396	83	446	301
samyk	0.0018	**0.0058**	2	554	125	176	209
D4Vince	**0.0066**	**0.0082**	7	608	499	165	187
n1nj4sec	**0.0058**	**0.0052**	8	876	1391	64	79

We find that the reciprocity is low and less than 7.3% for all the relationships in question. By contrast, reciprocity is often above 70% in social media, like Facebook or Twitter [12]. These social media mirror personal relationships and have an etiquette of conduct. We conjecture that the lower reciprocity on GitHub is due to its utilitarian orientation: following an author stems from a professional interest.

VI. Community Analysis

This section describes the functionality of the *community analysis* module, whose goal is to identify the communities of collaboration among the malware authors on GitHub.

A. Identifying collaboration communities. We quantify the collaborative nature of the malware authors as follows.

The Author-Repository graph (AR). We define the Author-Repository graph to be an undirected bipartite graph, $G = (A, R, E)$, where A is the set of malware authors and R is the set of malware repositories. An edge $(u, r) \in E$ exists, if author u: (a) creates, (b) stars, (c) forks, (d) watches, (e) comments, or (f) contributes to repository r.

Identifying bipartite communities. To identify communities, we employ a greedy modularity maximization algorithm modified for bipartite graphs as we discuss in our related work.

We find a total of 513 communities spanning a wide range of sizes as shown in Figure 5. The size of the communities follows skewed distribution. In Figure 5, we plot the number of malware authors and repositories per community in order of decreasing community size (defined as the sum of authors and repositories). We find that 90% of communities have less than 14 authors and repositories. We also see a fairly sharp knee in the plot at the fifth community, as shown by the vertical line.

B. Profiling the communities. A full investigation of the purpose, evolution, and internal structure of each community could be a research topic in its own right. Here, we only provide an initial investigation around the following three questions.

a. How cohesive are our communities? We report the **Modularity Score** ***(MS_C)***, which quantifies the cohesiveness of a community C. The MS_C is defined as follows: $MS_C = \frac{n_C(E)}{N_C(E)}$, where $n_C(E)$ is the total number of edges and $N_C(E)$

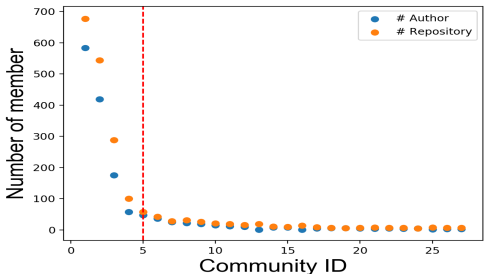

Figure 5: The distribution of the number of authors and repositories for the 27 largest communities in the order of community size.

is the number of all possible edges in community C (if the community was a bipartite clique).

Overall, **our communities are highly cohesive**: 82.8% (425) of the communities have a *Modularity Score* $MS_C \geq 50\%$, which means that more than half of all possible edges within the community exist. Interestingly, the largest communities exhibit strong cohesiveness. In Table III, we present a high-level profile of the five largest communities which have a Modularity Score of 0.65-0.78, which is indicative of tightly-connected communities.

b. Who are the community leaders? We want to identify the influential authors as part of profiling a community. We identify the top two most influential producers and connectors per community using the HackerScore from Section V. This leads us to a group of 144 leaders of the communities of size of at least 20 authors. We find 81% of these community leaders are part of the Highly Influential Group (HIG) of authors. This suggests that the HIG authors are indeed driving forces for the ecosystem. In the future, we intend to investigate in more depth the influence and dynamics of each community.

c. What is the focus of each community in terms of platform and malware type? A security expert would want to know the main type of malware (e.g. ransomware) and the target platform (e.g. Linux) of a community. We use the Repository Keyword Set, W_r, information of a repository r, as we defined in Section II, and we use it to characterize the community.

One way to quantify the importance of a keyword for a community is to measure the number of repositories, for which that keyword appears at least once. In detail, we use the **Strength Of Presence** *(SOP)* metric, which we define as follows. For a community C with a set of R repositories, we define k_i to be the number of repositories, in which keyword i appears in the metadata W_r for repository r at least once for all repositories $r \in R$. We define the SOP_i of keyword i from keyword set S as follows: $SOP_i = \frac{k_i}{\sum_{j \in S} k_j}$. In Table III, we show the most dominant keywords from malware types and platforms sets for each community and the related SOP scores.

We can also use the *SOP* to visualize the keywords as a word-cloud. A word-cloud is a more immediate, appealing, and visceral way to display the information. In Figure 6, we show the word-cloud for the third largest community, which is dominated by *ransomware* malware and targets *Windows* platforms. Not only we see the main words stand out, but their relative size conveys their dominance over the other words

Table III: High-level profile of the five largest communities of malware authors and malware repositories.

ID	Authors	Repos	MS	Dominant Platform	SOP	Dominant type	SOP
1	584	677	0.65	Linux	0.32	Keylogger	0.29
2	419	544	0.67	Windows	0.26	Virus	0.31
3	175	288	0.73	Windows	0.65	Ransomware	0.44
4	57	100	0.78	Linux	0.43	Spyware	0.43
5	47	57	0.71	Mac	0.33	Trojan	0.22

Figure 6: The word-cloud for the malware types and platforms keywords for the third largest community: Ransomware and Windows dominate.

more viscerally than a lengthy table of numbers.

We present the results of this type of profiling for the largest communities in Table III, which we also discuss below.

We find that the largest community of 584 malware authors and 677 malware repositories having Linux ($SOP = 0.32$) and keylogger ($SOP = 0.29$) as the dominant platform and malware type. Interestingly, we find that 49 of the top 100 most prolific (in terms of the number of repositories created) authors are in this community. Upon closer investigation, we find that 11 out of the 15 authors with the highest degree in the subgraph of this community are keylogger developers.

The third-largest community consists of 175 malware authors and 288 malware repositories and revolves around Ransomware ($SOP = 0.65$) and Windows platform ($SOP = 0.44$). For reference, we present the word-cloud of the malware types and platforms based on the *SOP* score in Figure 6 for this community which exhibits that Ransomware and Windows possess the highest SOP scores.

Finally, the fourth largest community (57 authors, 100 repositories) is the most tightly connected ($MS = 0.78$) and it revolves around the development of attack tools for Kali Linux. Upon closer inspection, we find that 15 of the top 25 authors (based on node degrees) form an approximate bipartite clique with 5 repositories. This group developed *WiFiPhisher* in 2016, a Linux-based python phishing tool [13], which has been used for both good and evil [14].

The above are indicative of the potential information that we could extract from these malware repositories. In the future, we intend to: (a) extract more detailed textual information from each community, and (b) study the evolution and dynamics of these communities over time.

VII. Author Investigation

"Who are these malware authors?" To answer this question, we go across platforms to security forums and leverage our datasets from several security forums. The functions described here are part of the *author investigation* module of Figure 2.

a. Malware authors strive for an online "brand" and

Table IV: Profiles of four cross-platform users.

Name	Forum	Posts in forum	Collab/tors in forums	Malw. repos	Follow-ers	Forks	Collab/tors in GitHub	Repository content	Internet-wide Reputation
misterch0c	WS	7	224	7	749	81	898	Cracked malware code	Self-declared hacker
3vilp4wn	HTS	103	513	1	0	1	6	Python keylogger	Keylogger developer
fahimmagsi	OC	73	175	1	1	0	1	Backdoor	Famous hacker
Evilcry	EH	18	444	2	89	15	98	Botnet and ransomware	Ransomware expert

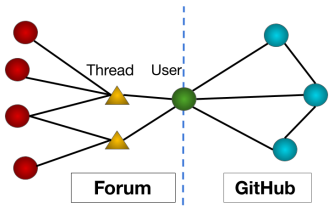

Figure 7: A cross-platform egonet: capturing the neighbors of both the security forum and GitHub.

usernames seem persistent across online platforms. We find that many malware authors use the same username consistently across many online platforms, such as security forums, possibly in pursuit of a reputation.

We identify 30 malware authors who are active in one of our four security forums: 12 in Wilders Security, 6 in Ethical Hacker, 4 in Offensive Community, and 8 in Hack This Site [10]. We argue that some of these usernames correspond to the same users based on the following two observations.

First, we find significant overlap in the interests of the cross-platform usernames. For example, usernames *int3grate* and *jedisct1* show interest in ransomware in both platforms, while *3vilp4wn* advertises her keylogger malware (github.com/3vilp4wn/CryptLog) in the forum. Second, these usernames are fairly uncommon, which increases the likelihood of belonging to the same person. For example, the top ten results from internet searching for the username of author *Misterch0c* returns nine hacker related sites and a twitter account with a different handle but claimed by Misterch0c. Note that not all the malware authors or repositories have a malicious purpose. For instance, the project "Empire" [15] by *xorrior* was created as an offensive tool to stress-test the security of systems. However, it has recently been used by the state-sponsored hacking group *Deep Panda* [16]. In general, offensive security tools contribute to the power of the malware ecosystem irrespective of the intention of its creator.

b. Modeling the cross-platform interactions. We propose to study the cross-platform interactions between GitHub and security forums as a promising research direction that can bridge two domains: software repositories and online forums.

We define the **cross-platform egonet** of a user as one that consists of her egonets from the two platforms as shown in Figure 7. The forum egonet captures the interaction of the users that post on the same threads, while we leverage the Author-Author network to define the GitHub egonet.

The value of cross-platform analysis. Using the cross-platform egonet as a basis, we can model the cross-platforms user dynamics, and more specifically, we can: (a) identify common "friends" between the ego-nets, (b) find the topics of interest and activities in each egonet, and (c) model information flow and influences across platforms. In Figure 1, we visualize the activity of a cross-platform user by comparing the number of users on each side of the egonet as shown in Figure 7. In Table IV, we show the actual values of indicative users, including the three outliers in the plot.

The cross-platform egonet analysis can enrich the profile of each user significantly. For example, if we were just looking at GitHub, we may not have paid attention to *3vilp4wn* and *Evilcry*. Both of these authors are less active on GitHub (small GitHub egonet), but are quite active in the security forums (large forum egonet). A closer investigation of the security forums reveals activities that match their interests on GitHub. This suggests that their GitHub activity is part of their online brand. For example, *3vilp4wn* advertises her GitHub keylogger repository in the forum.

c. Using information from the web. In our approach, we leverage existing information on hackers from (a) security outlets and databases, and (b) using web queries. With our python-based query and analysis tools, we verified the role and activities of authors, which we omit due to space limitations.

VIII. Related Works

Studying the dynamics of the malware ecosystem on GitHub has received very little attention. Most studies differ from our work in that: (a) they do not focus on malware on GitHub, and (b) when they do, they do not take an author-centric angle as we do here: they focus on classifying malware repositories or use a small set for a particular research study.

Our work builds on our earlier effort [2], whose main goal is to identify malware repositories on GitHub at scale, but it does not study the malware author ecosystem as we do here.

a. Studies of malware repositories on GitHub: Several other efforts have manually collected a small number of malware repositories with the purposes of a research study [17], [18]. Some other studies [5] [6] analyze malware source code from a software engineering perspective, but use only a small number of GitHub repositories as a reference.

b. Studies of benign repositories on GitHub: Many studies analyze benign repositories on GitHub from a point of view of software engineering or as a social network. Some efforts find influential users and analyze the motivation behind following, forking, and contributions [7], [11]. Earlier efforts study repositories by analyzing the repository-repository relationship graph [19], and by using an activity graph [20].

Several works in this area identify influential authors and repositories using: the starring activity [21], the Following-Star-Fork activity [22], or a rank-based approach [23]. Note that a version of the hyperlink-induced topic search algo-

rithm [24] has been used by some of the above efforts for calculating influence, but they do not adjust the weights to account for the different frequencies of the types of interactions between users.

For our bipartite clustering, we adapt the greedy modularity maximization approach [25][26].

c. Studies on security forums: This is a recent and less studied area of research. Most of the works focus on extracting entities of interest in security forums. An interesting study focuses on the dynamics of the black-market of hacking goods and services and their pricing [4]. Other studies focus on identifying important events and threats [8], [9], [27]. None of the aforementioned works focus on the dynamics among hackers across platforms.

d. Cross-platform study: Finally, some efforts study author activities on different software development forums, namely GitHub and Stack Overflow [28], [29], but do not consider information from security forums.

IX. Conclusion

We develop a systematic approach for studying the ecosystem of hackers. Our approach develops methods to identify (a) influential hackers, (b) communities of collaborating hackers, and (c) their cross-platform interactions. Our study concludes in three key takeaway messages: (a) the malware ecosystem is substantial and growing rapidly, (b) it is highly collaborative, and (c) it contains professional malicious hackers.

Our initial findings are just the beginning of a promising future effort that can shed light on this online malware author ecosystem, which spans software repositories and security forums. The current work thus can be seen as a building block that can enable new research directions.

Follow up research can expand on our work to develop preemptive security initiatives, such as: (a) monitoring hacker activity, (b) detecting emerging trends, and (c) identifying particularly influential hackers towards safeguarding the Internet.

X. Acknowledgement

This work was supported by the UC Multicampus-National Lab Collaborative Research and Training (UCNLCRT) award #LFR18548554.

References

[1] Aaron Holmes, "17 years old boy tried to hack twitter," https://www.businessinsider.com/twitter-hacker-florida-teen-past-minecraft-bitcoin-scams-2020-8/, August 2020.

[2] M. O. F. Rokon, R. Islam, A. Darki, E. E. Papalexakis, and M. Faloutsos, "Sourcefinder: Finding malware source-code from publicly available repositories in github," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*. USENIX, 2020, pp. 149–163.

[3] J. Gharibshah, E. E. Papalexakis, and M. Faloutsos, "REST: A thread embedding approach for identifying and classifying user-specified information in security forums." *ICWSM*, 2020.

[4] R. S. Portnoff, S. Afroz, G. Durrett, J. K. Kummerfeld, T. Berg-Kirkpatrick, D. McCoy, K. Levchenko, and V. Paxson, "Tools for automated analysis of cybercriminal markets," in *WWW*, 2017, p. 657.

[5] A. Calleja, J. Tapiador, and J. Caballero, "A look into 30 years of malware development from a software metrics perspective," in *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 2016, pp. 325–345.

[6] A. Calleja, J. Tapiador, and J. Cabalero, "The malsource dataset: Quantifying complexity and code reuse in malware development," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 12, pp. 3175–3190, 2018.

[7] K. Blincoe, J. Sheoran, S. Goggins, E. Petakovic, and D. Damian, "Understanding the popular users: Following, affiliation influence and leadership on github," *Information and Software Technology*, vol. 70, pp. 30–39, 2016.

[8] A. Sapienza, A. Bessi, S. Damodaran, P. Shakarian, K. Lerman, and E. Ferrara, "Early warnings of cyber threats in online discussions," in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, Nov 2017, pp. 667–674.

[9] A. Sapienza, S. K. Ernala, A. Bessi, K. Lerman, and E. Ferrara, "Discover: Mining online chatter for emerging cyber threats," in *Companion Proceedings of the The Web Conference 2018*.

[10] Security Forums, "Ethical hacker, hack this site, offensive community, wilders security," https://www.ethicalhacker.net/, https://www.hackthissite.org/, http://offensivecommunity.net/, https://www.wilderssecurity.com/.

[11] J. Jiang, D. Lo, J. He, X. Xia, P. S. Kochhar, and L. Zhang, "Why and how developers fork what from whom in github," *Empirical Software Engineering*, vol. 22, no. 1, pp. 547–578, 2017.

[12] J. Weng, E.-P. Lim, J. Jiang, and Q. He, "Twitterrank: finding topic-sensitive influential twitterers," in *Proceedings of the third ACM international conference on Web search and data mining*, 2010, pp. 261–270.

[13] Sophron, "Wifiphisher," https://github.com/wifiphisher/wifiphisher, 2014, [Online; accessed 14-March-2020].

[14] Cybersec, "Stealing password in 5 minutes using wifiphisher," https://www.secjuice.com/phishing-with-wifiphisher/, 2018.

[15] EmpireProject, "Project empire," https://github.com/EmpireProject/Empire.

[16] Mitre, "State sponsored hacking tool," https://attack.mitre.org/software/S0363, 2019.

[17] T. Lepik, K. Maennel, M. Ernits, and O. Maennel, "Art and automation of teaching malware reverse engineering," in *International Conference on Learning and Collaboration Technologies*. Springer, 2018, pp. 461–472.

[18] X. Zhong, Y. Fu, L. Yu, R. Brooks, and G. K. Venayagamoorthy, "Stealthy malware traffic-not as innocent as it looks," in *2015 10th International Conference on Malicious and Unwanted Software*. IEEE, 2015, pp. 110–116.

[19] F. Thung, T. F. Bissyande, D. Lo, and L. Jiang, "Network structure of social coding in github," in *2013 17th European conference on software maintenance and reengineering*. IEEE, 2013, pp. 323–326.

[20] J. Xavier, A. Macedo, and M. de Almeida Maia, "Understanding the popularity of reporters and assignees in the github." in *SEKE*, 2014.

[21] Y. Hu, J. Zhang, X. Bai, S. Yu, and Z. Yang, "Influence analysis of github repositories," *SpringerPlus*, vol. 5, no. 1, pp. 1–19, 2016.

[22] Y. Hu, S. Wang, Y. Ren, and K.-K. R. Choo, "User influence analysis for github developer social networks," *Expert Systems with Applications*, vol. 108, pp. 108–118, 2018.

[23] Z. Liao, H. Jin, Y. Li, B. Zhao, J. Wu, and S. Liu, "Devrank: Mining influential developers in github," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[24] L. Li, Y. Shang, and W. Zhang, "Improvement of hits-based algorithms on web documents," in *Proceedings of the 11th international conference on World Wide Web*, 2002, pp. 527–535.

[25] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Physical review E*, vol. 70, no. 6, p. 6, 2004.

[26] T. Alzahrani and K. J. Horadam, "Community detection in bipartite networks: Algorithms and case studies," in *Complex systems and networks*. Springer, 2016, pp. 25–50.

[27] R. Islam, M. O. F. Rokon, E. E. Papalexakis, and M. Faloutsos, "Tenfor: A tensor-based tool to extract interestingevents from security forums," in *Proceedings of International Conference on Advances in Social Network Analysis and Mining (ASONAM)*. IEEE/ACM, 2020.

[28] C. Hauff and G. Gousios, "Matching github developer profiles to job advertisements," in *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*. IEEE, 2015, pp. 362–366.

[29] R. K.-W. Lee and D. Lo, "Github and stack overflow: Analyzing developer interests across multiple social collaborative platforms," in *International Conference on Social Informatics*. Springer, 2017, pp. 245–256.