Six Million (Suspected) Fake Stars on GitHub: A Growing Spiral of Popularity Contests, Spam, and Malware

Hao He
Carnegie Mellon University
Pittsburgh, PA, USA
haohe@andrew.cmu.edu

Haoqin Yang
Carnegie Mellon University
Pittsburgh, PA, USA
haoqiny@andrew.cmu.edu

Philipp Burckhardt
Socket Inc
Pittsburgh, PA, USA
philipp@socket.dev

Alexandros Kapravelos
North Carolina State University
Raleigh, NC, USA
akaprav@ncsu.edu

Bogdan Vasilescu
Carnegie Mellon University
Pittsburgh, PA, USA
vasilescu@cmu.edu

Christian Kästner
Carnegie Mellon University
Pittsburgh, PA, USA
kaestner@cs.cmu.edu

Abstract

GitHub, the de facto platform for open-source software development, provides a set of social-media-like features to signal high-quality repositories. Among them, the star count is the most widely used popularity signal, but it is also at risk of being artificially inflated (i.e., *faked*), decreasing its value as a decision-making signal and posing a security risk to all GitHub users. In this paper, we present a systematic, global, and longitudinal measurement study of fake stars in GitHub. To this end, we build StarScout, a scalable tool able to detect anomalous starring behaviors across all GitHub metadata between 2019 and 2024. Analyzing the data collected using StarScout, we find that: (1) fake-star-related activities have rapidly surged in 2024; (2) the accounts and repositories in fake star campaigns have highly trivial activity patterns; (3) the majority of fake stars are used to promote short-lived phishing malware repositories; the remaining ones are mostly used to promote AI/LLM, blockchain, tool/application, and tutorial/demo repositories; (4) while repositories may have acquired fake stars for growth hacking, fake stars only have a promotion effect in the short term (i.e., less than two months) and become a liability in the long term. Our study has implications for platform moderators, open-source practitioners, and supply chain security researchers.

CCS Concepts

• **Human-centered computing** → *Open source software*; *Reputation systems*; • **Security and privacy** → *Web application security*.

Keywords

GitHub, supply chain security, fake stars, longitudinal analysis

ACM Reference Format:
Hao He, Haoqin Yang, Philipp Burckhardt, Alexandros Kapravelos, Bogdan Vasilescu, and Christian Kästner. 2026. Six Million (Suspected) Fake Stars on GitHub: A Growing Spiral of Popularity Contests, Spam, and Malware. In *2026 IEEE/ACM 48th International Conference on Software Engineering (ICSE '26), April 12–18, 2026, Rio de Janeiro, Brazil.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3744916.3764531

ICSE '26, Rio de Janeiro, Brazil

ACM ISBN 979-8-4007-2025-3/26/04
https://doi.org/10.1145/3744916.3764531

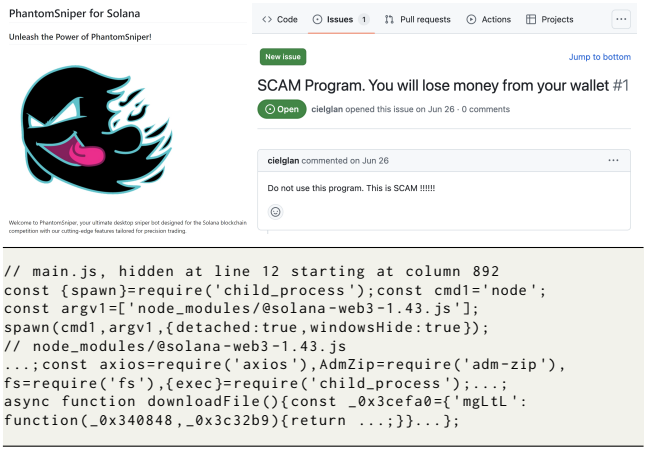


Figure 1: An example malware repository detected by our tool StarScout, reported to GitHub, and since deleted. It had 111 stars, of which we suspect at least 109 to be fake. The README file (top left) suggests a blockchain application, but if executed, its code (bottom) steals cryptocurrencies using a hidden `spawn()` call to a heavily obfuscated remote file execution script (with the name of a seemingly legitimate JavaScript package). An issue thread (top right), presumably created by a victim, warns of malware hidden inside.

1 Introduction

> *The more any quantitative social indicator is used for social decision-making, the more subject it will be to corruption pressures and the more apt it will be to distort and corrupt the social processes it is intended to monitor.*
> - Donald T. Campbell [33]

Many studies provided evidence that the number of GitHub stars is widely used when stakeholders evaluate, select, and adopt open-source projects into their software supply chain [32, 60, 68, 73, 86, 92]. However, it is merely a popularity signal (i.e., the attention received from GitHub users), with only moderate correlation to actual usage or importance as shown in prior studies [55, 60, 77].

More concerning, *GitHub stars can be artificially inflated* [13], just as all other popularity signals in social media [36]. For example, a Google search for "buy GitHub stars" will reveal a dozen providers

(see Appendix, Section 6). From prior reports, we know that GitHub repositories may buy GitHub stars for growth hacking [3], spamming [10], résumé fraud [43], or even for spreading malware [21]. Whatever their purpose, these bought GitHub stars (hereafter "fake stars" to match online discussions) corrupt the GitHub star count as a decision-making signal and pose a security threat to all GitHub users (see Figure 1 for an example).

However, most prior reports of fake GitHub stars come from the gray literature (i.e., non-peer-reviewed blog posts, news reports, and online discussions), and they do not go beyond studying a few specific cases [3, 10, 21, 43]. With the rising prevalence of software supply chain attacks [17], we believe it is important to conduct *a systematic, global, and longitudinal measurement of this emerging threat, especially from the lens of software supply chain security*. This would contribute to our understanding of fake GitHub stars and the fraudulent/malicious activities around them, furthering the design of countermeasures and informing best practices on the usage of an important software supply chain metric.

Several challenges persist in the global measurement of fake GitHub stars. First, GitHub stars can be faked in various ways (e.g., bots [21], crowdsourced humans [3], exchange platforms where users exchange stars for a reward [43]), and malicious actors continuously evolve their behaviors to evade platform detection [39, 57]. Their actions blur the boundary between fake and authentic stars, posing a challenge to the identification and measurement of fake ones. Second, the sheer amount of GitHub data (~20TB of metadata in the past five years [1]) and the rate limit of GitHub APIs [20] present challenges to a global analysis. Finally, GitHub does not preserve deleted repositories and users, posing difficulties in the measurement of fraudulent or malicious activities, as some of them may have already been taken down by the platform.

To overcome these challenges, we take advantage of prior work on mining software repositories [1] and social media fraud detection [29, 39] to implement StarScout, a scalable tool that detects (suspected) *fake stars* over the entire GHArchive [1] dataset, a Google BigQuery replica of all GitHub events (totaling 63.9 million users, 331 million repositories, 326 million stars, and 6.7 billion other events as of January 2025). StarScout (details in Section 3) identifies two signatures of anomalous starring behaviors, *the low activity signature* and *the lockstep signature*, and includes a post-processing step to reduce noise and increase overall accuracy.[1]

We apply StarScout to all GitHub event data from July 2019 to December 2024, identifying *6.0 million fake stars and 18,617 repositories with fake star campaigns*. Our evaluation shows that: (a) StarScout can detect 81% of repositories and 76% of accounts in a confirmed malware campaign involving fake stars [21]; (b) the detected repositories and accounts have anomalously high deletion ratio, up to 90%, 16x higher compared to random GitHub repositories and accounts.

Using the dataset, we further conduct a measurement study that answers the following research questions:

- **RQ1** (Prevalance): *How prevalent are fake GitHub stars?*
- **RQ2** (Activity Patterns): *What are the activity patterns of GitHub repositories and accounts with fake star campaigns?*
- **RQ3** (Repository Charactersitics): *What are the GitHub repositories with fake star campaigns?*
- **RQ4** (Promotional Effect): *To what extent are fake stars effective in promoting the target GitHub repositories?*

Our findings paint an alarming picture: Fake stars have surged in 2024, either to promote spam or phishing malware repositories, or fuel popularity contests in hyped domains (e.g., blockchain and AI). In the latter case, we show that fake stars, while increasing the displayed star counts in the short term, fail to bring true attention in the long term. From a security perspective, our results call for future research into malicious activities happening on GitHub (e.g., fake stars, fake accounts, spam, phishing). From a practitioner's perspective, our results further demonstrate the limitation of star counts as a popularity signal, provide counter-evidence to the effectiveness of fake stars for growth hacking, and call for the design of better popularity signals for GitHub repositories.

In summary, this paper makes the following contributions:

- We design and implement StarScout, a scalable tool to scan for anomalous starring behavior across all of GitHub. This leads to a novel large-scale dataset regarding fake GitHub stars.
- Our measurement study discovers novel findings regarding the prevalence, characteristics, and effectiveness of fake stars, leading to practical insights for platform moderators, open-source practitioners, and supply chain security researchers.

2 Background and Related Work

GitHub is the de facto platform where developers collaborate in open-source projects (organized as *repositories*). It is designed to be transparent, and repositories can build their reputation through technical and social signals (e.g., stars, forks, badges) [84, 85]. As the design of these social signals resembles those in online social networks, we first introduce spam/fraud research there (Section 2.1). Then, we introduce software supply chain attacks (Section 2.2) and summarize prior findings on (fake) GitHub stars (Section 2.3).

2.1 Fraudulent Activities in Social Networks

In a nutshell, most of the fraudulent activities in online social networks are about controlling a group of fake accounts to spread spam or generate fake signals. There is a rich literature on countering fraudulent activities in online social networks (see, e.g., recent literature reviews [26, 63, 74]). For example, researchers have studied how fake accounts can be used to spread advertisements [83], malware [50], misinformation [79], and political opinions [49, 72]. They can also be used to artificially boost the popularity of posts or users (using, e.g., fake Likes in Facebook [29, 57] and Instagram [78], and fake followers and retweets in Twitter [80, 81]). For the latter scenario, a recurring observation from prior research is that some providers generate fake popularity signals using automated bots (aka. sybils), but some others use realistic accounts and even crowdsourced real humans (aka. crowdturfing) [57, 80, 81, 90, 97].

With a ground truth dataset and reasonable feature engineering, it is often possible to build high accuracy supervised machine learning models to detect such activities [e.g., 80, 81, 95], but these models can be evaded by malicious actors if they change their behaviors over time [39, 89]. Another line of research uses unsupervised techniques to detect anomalous users and activity clusters

[1]Note that technically speaking, StarScout only detects *suspected* fake stars and *suspected* fake star campaigns, among which there may still be false positives (see Section 3.5 for a discussion). However, to make this paper more readable, we will simply refer to them as *fake stars* and *fake star campaigns* in the remainder of this paper.

that deviate significantly from normal ones [29, 35, 56, 87]. However, Ikram et al. [57] show that the Facebook Like farms are still able to maneuver through these techniques by mimicking realistic user activities and violating the techniques' built-in assumptions (e.g., assuming that fake Likes happen in bursts [29]). In general, the battle against fraudulent activities in social media is a never-ending arms race [39, 96]. Our study contributes to this literature by providing a systematic report of an emerging fraudulent activity (i.e., fake stars) in a novel setting (i.e., GitHub, the de facto social coding platform for open-source projects).

2.2 Software Supply Chain Attacks

Modern software development is powered by a collaboratively developed digital infrastructure of open-source software components [45], most of which are maintained on GitHub and published in package registries (e.g., npm and PyPI). In recent years, malicious actors have been trying to exploit both the open-source components and the platforms hosting them, creating an emerging attack vector, often referred to as *software supply chain attacks* (see recent literature reviews [62, 69] for more details). The end goal of these attacks is to sneak malware in a software project, through which attackers can compromise production systems [93], inject backdoors [11], steal cryptocurrencies [7], etc.

One common type of attack is to compromise existing open-source components and inject malware into them. This type of attack can be extremely impactful as open-source components form complex dependency networks in which one component may end up being depended upon by a huge number of downstream applications [41, 100]. Alternatively, attackers can create new, seemingly useful, but malicious software, and try to get them adopted. Examples of this kind of attack include typosquatting in package registries [88], publishing malicious VS Code extensions [6], and phishing in GitHub repositories [21, 34]. Fake stars can be a useful weapon for both kinds of attacks: An attacker can either establish a fake reputation and seek maintainership of critical open-source projects or buy fake stars to promote their own phishing repositories. Our study aims to reveal the achieved or potential role of fake stars in (possibly ongoing) software supply chain attacks.

2.3 GitHub Stars

Similar to Likes in social networks, GitHub provides a "Star" button for users to show appreciation or bookmark a GitHub repository [28, 32]. The widespread use of this feature makes the number of stars a widely recognized popularity signal for GitHub repositories, e.g., is often used for open-source project selection by both practitioners [73, 86, 92] and researchers [60, 68]. However, researchers have also shown that the number of GitHub stars does not correlate highly with the number of downloads and actual installations [60] and does not predict the importance or sustainability of an open-source project [55, 77]. Despite studies arguing the limited reliability of GitHub stars, the lack of obvious alternatives makes it still an important signal in these decision-making processes.

The GitHub Star Black Market. The widespread use of GitHub stars catalyzed the emergence of a GitHub star black market [13], which operates similarly to other black markets in social media that sell Twitter followers [81], Facebook likes [29], etc. From the gray literature, we know that the GitHub star black market operates in at least three different ways: (1) Merchants can sell GitHub stars in batches of (usually) 50 or 100 stars on their own websites (examples in Appendix, Section 6), instant messaging apps [43], or in e-commerce platforms such as Taobao [4]; (2) GitHub users may form exchange platforms (e.g., GitStar [14] or instant messaging groups), where they incentivize mutual starring of their GitHub repositories [43]; (3) A GitHub repository may directly incentivize the audience of its advertising campaign with gifts if they star the repository (e.g., as it happened in OceanBase [3]). All these ways of operation violate the GitHub Acceptable Use Policy [12], which prohibits any inauthentic interactions, rank abuse, and reward-incentivized activities. In all three cases discussed above, we consider these purchased, exchanged, or incentivized GitHub stars as *fake* because they are artificially created and do not genuinely represent any authentic appreciations, uses, or bookmarks of a repository from real GitHub users.

Prior Work on Fake GitHub Stars. As the first and only academic report on this emerging black market (to the best of our knowledge), Du et al. [43] collected 1,023 black market GitHub accounts through honeypots and built machine learning classifiers to identify 63,872 more suspected GitHub accounts from 2015 to 2019. They further conducted a characterization study of these suspected black market fake accounts, but they did not conduct any measurement or characterization of *repositories with fake GitHub stars*. On the other hand, the gray literature (i.e., non-peer-reviewed news reports, blog posts, and online discussions) provides evidence on how and why repositories may fake GitHub stars. For example, repositories may fake GitHub stars for growth hacking [31] (i.e., *"fake it until you make it"*). Startups (and occasionally, even large companies) buy GitHub stars to promote their open-source products [3, 10] because VCs (and managers) refer to GitHub stars for product viability [9]. Another reported reason for faking GitHub stars is résumé fraud, where software developers may want to fake GitHub profiles to increase their competitiveness in hiring [43]. Finally, malicious actors may fake GitHub stars to promote repositories with malware [21, 82] (e.g., Figure 1). These reports plot an alarming rising threat to software supply chain security: Repositories with fake stars may gain an unfair advantage in the GitHub popularity contest, which can be then exploited in various ways to harm stakeholders in the software supply chain. To counter this rising threat, it is vital to have a thorough and systematic investigation of fake GitHub stars and the repositories around them. This forms the main motivation of our study.

3 StarScout Design

3.1 Overview

At a high level, StarScout finds two signatures of anomalous starring behaviors from GitHub stargazer history: the *low activity signature* and the *lockstep signature*. Both signatures are hard to avoid for accounts controlled by GitHub star merchants: Whatever obfuscation methods they use or however realistic they are, these accounts must be either newly-registered throw-away accounts or have been synchronously starring repositories in short time windows to meet their promised delivery times. Specifically, the low-activity signature identifies stars coming from accounts that

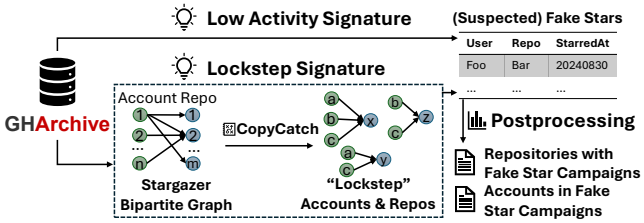


Figure 2: A high-level overview of StarScout.

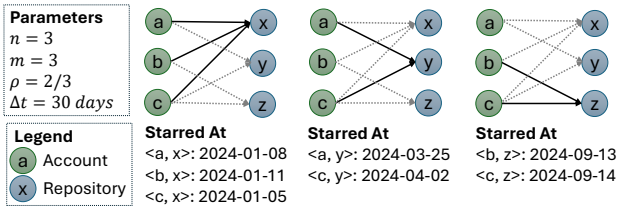


Figure 3: Under the parameter setting shown above, accounts a, b, c and repositories x, y, z are considered to demonstrate a lockstep signature because each repository has stars from at least two of these accounts in a 30-day time window.

become stale after merely starring one (or a few) repositories; and the lockstep signature identifies stars coming from clusters of n accounts that have been repeatedly acting together to star another cluster of m repositories in short Δt time windows. However, both signatures may lead to false positives (both to repositories and to accounts), as a legitimate user may happen to behave similarly, or a fake account may star legitimate repositories to hide themselves. Therefore, StarScout applies an additional postprocessing step to identify repositories and accounts with fake star campaigns. We provide an overview of StarScout in Figure 2, and we will describe StarScout in more detail in the remainder of this section.

3.2 Heuristics and Postprocessing

The Low Activity Signature. To detect the low activity signature, StarScout finds GitHub accounts with only one `WatchEvent` (i.e., have only starred one GitHub repository), plus at most one additional event (e.g., `ForkEvent`) in the same repository on the same day. This heuristic is inspired by a heuristic proposed in the gray literature [10], but further simplified and tuned to avoid costly interactions with the rate-limited GitHub APIs and easy-to-obfuscate rules. While the detected accounts may be throw-away bot accounts controlled by fake star merchants, they may also be real users wrongly identified (e.g., someone legitimately registered an account, starred a repository, and then gave up using GitHub). To partially address such cases, we also look at fake star counts at the repository level. That is, we only consider repositories with at least 50 stars suspected as fake, based on the observation that fake star merchants often sell a minimum of 50 stars (see Appendix, Section 6). Consequently, we ignore the fake stars (and corresponding users) in repositories with fewer than 50 fake stars in total. We will also use this cutoff of 50 in the lockstep signature detector and the postprocessing step, based on the same intuition.

The Lockstep Signature. If a fake star merchant controls a group of accounts to repeatedly deliver stars to repositories in their promised delivery time, it would create a unique "lockstep" signature. Prior social network research shows that this signature is unlikely to form naturally and highly correlated to fraudulent activities [29, 56, 70]. Mathematically speaking, let $\mathbf{U}$ be the set of all GitHub accounts and $\mathbf{R}$ be the set of all GitHub repositories; all GitHub stars effectively form a bipartite graph $\langle \mathbf{U}, \mathbf{R}, \mathbf{E} \subseteq \mathbf{U} \times \mathbf{R}, \tau : \mathbf{E} \mapsto \mathbf{T} \rangle$ where each $\langle u, r \rangle \in \mathbf{E}$ represents a star from account u to repository r and $\tau(\langle u, r \rangle)$ maps to the time of starring. Given four parameters $\langle n, m, \rho, \Delta t \rangle$, we consider a group of accounts $U \subseteq \mathbf{U}$ and repositories $R \subseteq \mathbf{R}$ to demonstrate a *lockstep signature* if:

$$\begin{aligned} & |U| \geq n, \quad |R| \geq m \\ \forall r \in R, \exists U_r \subseteq U, \exists t_r \in \mathbf{T}: \quad & |U_r| \geq \rho n, \\ & \forall u \in U_r : \langle u, r \rangle \in \mathbf{E}, \\ & \forall u \in U_r : \tau(\langle u, r \rangle) \in [t_r, t_r + \Delta t] \end{aligned}$$

In other words, we consider a group of accounts and repositories to be "lockstepping" if each repository has stars from at least ρn accounts in a time window no longer than Δt (see Figure 3 for an example). Note that this definition is both an extension and a relaxation of bicliques (or complete bipartite subgraphs) [71], as the latter are generally very rare in real large-scale bipartite graphs.

To find accounts and repositories with the lockstep signature at scale, StarScout implements CopyCatch [29], a state-of-the-art algorithm that has been deployed at Facebook to detect fake Likes with the same lockstep signatures. At a high level, the algorithm starts from a set of seed repositories. For each seed repository, it iteratively calibrates a time window and finds the largest possible group of accounts and repositories demonstrating a lockstep signature within that time window. Finally, groups with more than n accounts and m repositories are retained and considered as engaging in suspicious fake starring activities. We refer interested readers to the original paper [29] and our implementation (Section 6) for the details of CopyCatch. While the problem of finding maximum bicliques is NP-complete [71] and CopyCatch is fundamentally a greedy local search algorithm, it is highly scalable and has been shown to work reasonably well in practice [29].

Postprocessing. Even if the above two signatures identify significantly anomalous patterns in GitHub stargazer data, it is unreasonable to expect that every repository receiving fake stars is actively seeking fake stars. In fact, we find that many highly popular repositories get a relatively small amount of fake stars; we infer that fake accounts deliberately star them as an adversarial behavior to evade platform detection. Therefore, *the postprocessing step aims to identify and keep only repositories with an anomalous spike of fake stars that contributes a non-negligible portion of stars obtained in that repository.* We infer that these repositories have probably run a fake star campaign and are probably not a victim of fake stargazers, as they gained a noticeable benefit from those stars. For this purpose, StarScout aggregates monthly star counts to find repositories where (1) there is at least one single month in which it gets more than 50 fake stars and the percentage of fake stars exceeds 50%;[2] (2) the all-time percentage of fake stars (relative

[2]The magic number 50 follows the same rationale as the low activity signature, that fake star merchants often sell a minimum of 50 stars (see Appendix, Section 6).

to all stars) exceeds 10%. StarScout considers these repositories as *repositories with fake star campaigns* and the accounts that starred in these anomalous spike months as *accounts in fake star campaigns*.

3.3 Implementation

We implement detectors for the low activity signature and each iteration step of CopyCatch (for detecting the lockstep signature) as SQL queries on Google BigQuery, which enables StarScout to scale the detection to all GitHub in a distributed manner. We use Python scripts to dynamically generate and send these queries, and collect fake stars from each query output into a local MongoDB database (using Google Cloud Storage as relay). We set the lockstep signature parameters as: $n = 50$, $m = 10$, $\Delta t = 30$ *days*, and $\rho = 0.5$; in other words, we seek to find clusters of ≥50 accounts and ≥10 repositories, such that there exists a 30-day interval for each repository in which it gets ≥25 stars from these accounts. As the stargazer bipartite graph is extremely large (326 million edges from July 2019 to December 2024), we split the graph into interleaving six-month chunks (e.g., Jan - Jun 2024, Apr - Sept 2024). The chunks are interleaved to avoid missing detections across chunk boundaries, and new chunks can be analyzed on a quarterly basis.

We initially deployed StarScout in July 2024 when it scanned fake stars in the past five years, totaling more than 20 TB of data from GHArchive. Then, we updated and merged two additional chunks with the latest cutoff of December 2024. In total, StarScout identified **six million (suspected) fake stars** across 26,254 repositories before the postprocessing step; among these stars, 1.06 million are identified with the low activity signature and 4.93 million with the lockstep signature. In the postprocessing step, StarScout identified 18,617 repositories with fake star campaigns and 301k participating accounts (corresponding to 3.81 million fake stars).

3.4 Evaluation

It is challenging to evaluate a fake activity detector like StarScout. While honeypots are often used in prior research (e.g., [38, 40, 43]), they raise ethical concerns [61, 91]: If we were to buy fake stars for a honeypot repository, we would be effectively funding malicious actors and creating spam for other developers. To alleviate this concern, we evaluate the recall of StarScout using an alternative ground truth malware phishing campaign arguably powered by fake stars. Specifically, we use the Stargazer Ghost Network dataset [21], which contains 847 repositories with more than 50 GitHub stars coming from 15,672 highly suspicious GitHub accounts. We find that StarScout can detect 688 (81.23%) of the 847 repositories and 11,903 (75.95%) of the 15,672 involved GitHub accounts. *In other words, StarScout can achieve up to 82.23% recall for repositories and 75.95% recall for accounts when it comes to the detection of fake-star-powered malware campaigns.* We believe this recall performance is satisfactory, especially considering that the algorithm we use for detecting the lockstep signature, CopyCatch, is a randomized greedy algorithm with no completeness guarantees [29].

Precision is inherently harder to evaluate because there is no obvious way to examine whether a star is fake or not. Instead, we estimate *criterion validity* [37], i.e., the extent to which an outcome is correlated to some other theoretically-relevant outcomes (or criterion). In our case, we compare *the percentage of GitHub repositories and accounts that have been deleted* (as of January 2025) between a comparison sample and the sample detected by StarScout. We choose this criterion based on reports of GitHub actively countering fraudulent and malicious activities [13]—thus, we should be able to observe a higher deletion rate in the StarScout sample compared to the baseline, especially if fake stars are also used for other malicious activities (like the Stargazer Ghost Network [21]). Therefore, for all repositories and 10,000 sample accounts (downsampled due to GitHub's API rate limit) found by StarScout, we check (using the GitHub API) whether they were still accessible at the time of detection. We also randomly sampled another 10,000 GitHub repositories with ≥50 stars and 10,000 GitHub stargazers in these repositories, to get baseline deletion percentages for comparison.

Table 1: The fraction of repositories/accounts detected by StarScout and deleted on GitHub at the time of detection is drastically higher compared to the baseline level obtained from random GitHub repositories and users.

	% Deleted	
	w/o Postprocessing	w. Postprocessing
Repositories (Baseline: 5.03%)		
Low Activity	14.38%	79.36%
Lockstep	82.03%	90.70%
All	**70.05%**	**90.42%**
Accounts (Baseline: 3.54%)		
Low Activity	19.19%	72.29%
Lockstep	23.03%	48.83%
All	**18.77%**	**57.07%**

Compared with the baseline deletion ratio (5.03% for repositories and 3.54% for users), the deletion ratios for the detected repositories and accounts are substantially higher (Table 1): 90.42% of repositories and 57.07% of accounts in fake star campaigns (i.e., with postprocessing in Table 1) have been deleted. The percentage is lower if we consider all repositories and accounts with fake stars (i.e., without postprocessing in Table 1), especially for those detected by the low activity signature, providing support for the effectiveness and necessity of the postprocessing step. Overall, the results align with our intuition that both signatures may be noisy in identifying individual fake stars but, when combined, effective in identifying repositories with fake star campaigns. The deletion ratio of accounts is generally lower than that of repositories, which can be interpreted in multiple ways. For example, it may be that StarScout is less accurate at identifying sophisticated fake accounts (e.g., those detected by the lockstep signature); it is also possible that GitHub prioritizes taking down malware/phishing repositories, as supported by our RQ3 results (Section 4.3).

3.5 Limitations and Ethical Concerns

The most important limitation of StarScout is that, while it detects statistically anomalous starring patterns in GitHub, the evidence from StarScout alone is insufficient for disciplinary action (e.g., for taking down fraudulent repositories and accounts). The latter would usually require evidence from non-public information (e.g., IP addresses). In general, it is possible that some of the StarScout

identified repositories and accounts may still be false positives. Another limitation of StarScout is the use of ad-hoc heuristics and parameters (notably the 50 stars threshold in many parts of detection). While it is challenging to run thorough sensitivity tests on these parameters due to the sheer amount of data to be processed, we used our best judgment in choosing these parameters. Also, considering the generally high face validity of our evaluation and measurement study results (Section 3.4 and Section 4), we believe the performance of StarScout suffices for our research goal—characterizing the landscape of an emerging threat.

The possibility of false positives in StarScout detections also brings ethical concerns. Even if StarScout is ~99% precise in detecting repositories that bought fake stars, there would still be ~180 false positive repositories in our resulting dataset, and interpreting them as having done so would potentially harm their reputation. To mitigate such risks, we focus on presenting statistical patterns and avoid giving individual repositories as examples in the paper (unless they are obviously malicious). We also carefully discussed and noted ethical concerns in the dataset documentation, calling for its responsible use and interpretation (Section 6).

4 Measurement Study

Using the dataset of 18,617 repositories and 301k accounts with fake star campaigns (Section 3.3), we conduct a measurement study of fraudulent starring activities in GitHub, that answers the following four research questions (formulated in Section 1):

- **RQ1** (Prevalance): *How prevalent are fake GitHub stars?*
- **RQ2** (Activity Patterns): *What are the activity patterns of GitHub repositories and accounts with fake star campaigns?*
- **RQ3** (Repository Characersitics): *What are the GitHub repositories with fake star campaigns?*
- **RQ4** (Promotional Effect): *To what extent are fake stars effective in promoting the target GitHub repositories?*

The first three research questions involve different angles of exploratory analysis of this dataset. The final research question aims to quantify the impact of fake star campaigns: Are they really effective in attracting more (legitimate) attention just as real stars, or only effective in making repositories seem momentarily popular?

4.1 RQ1: Prevalence

Motivation. While StarScout has detected an enormous number of repositories and accounts with fake star campaigns, it is not yet clear to what extent these repositories and accounts may affect open-source software development and the software supply chain as a whole. To understand this, we need to provide evidence on: (a) the prevalence of fake stars and involved repositories & accounts relative to regular GitHub activities; (b) the prevalence of fake-star-related repositories in popular distribution channels (e.g., GitHub Trending [22] and package registries such as npm and PyPI).

Prevalence w.r.t. Overall GitHub Activity. For each month in our observation period (July 2019 to December 2024), we compute the following: (a) the percentage of fake GitHub stars among all GitHub stars in that month; (b) the percentage of GitHub accounts in fake star campaigns, among all active GitHub users (i.e., at least one GHArchive event) in that month; (c) the percentage of GitHub repositories with fake star campaigns among all repositories that received ≥ 50 stars in that month. Through this operationalization, we can estimate the portion of GitHub activities that may have been faked and the impact of these faked activities.

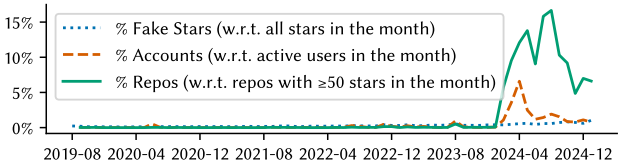


Figure 4: The percentage (%) of GitHub stars, accounts, and repositories involved in fake star campaigns in each month.

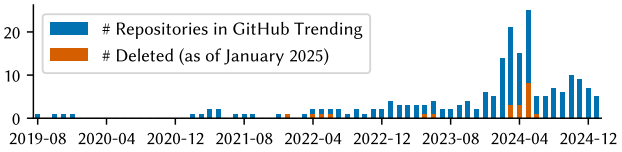


Figure 5: The number (#) of GitHub repositories with fake star campaigns and appeared in GitHub Trending each month.

We observe in Figure 4 that fake stars still only constitute a small fraction (≤1%) of all GitHub stars each month, but fake star campaigns have been increasing since 2022 and surged since 2024. Their broadest impact was in July 2024, when 16.66% of popular repositories (3,499) had fake star campaigns. March 2024 sees the highest percentage of fake star accounts (6.59%, 117,024).

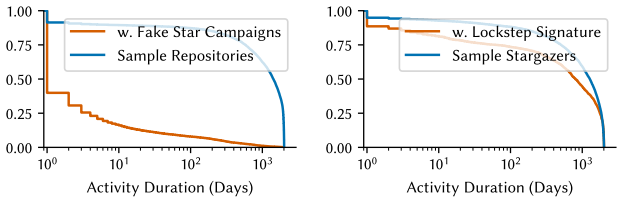
Finding 1: Fraudulent starring activities started to gain momentum in 2022 and have surged in 2024.

Prevalence in GitHub Trending. While the above analysis provides evidence regarding the general prevalence of repositories with fake star campaigns, it is still unclear whether these repositories actually reach developers. To study this, we pull historical records of GitHub Trending [22] repositories from a community-maintained archive [25]. Note that this archive only collects trending repositories from five programming languages (Python, Go, C++, JavaScript, and CoffeeScript), so we likely underestimate the actual reach of the studied repositories in GitHub Trending. In total, we identified 78 (0.42%) repositories with fake star campaigns that have also appeared in GitHub Trending. While we also see a similar 2024 surge (Figure 5), the peak number is only 25 (in March 2024). This lower reach, compared with the results in Figure 4, suggests that the GitHub trending algorithm is effective at filtering out most superficially-popular repositories. Among the matched subset, we anecdotally note repository names suggestive of malicious activities, e.g., `Wallet-stealer` and `blooket-hacks`; we will further analyze these repositories in Section 4.3.

Finding 2: Only a small number (78; 0.42%) of repositories with fake star campaigns reached the GitHub Trending page.

Table 2: Descriptive statistics for the 738 packages linked to 229 repositories with fake star campaigns.

	mean	min	25%	50%	75%	max
# stars	1686.69	54	240	380	1235	29.06k
# dependent packages	1.69	0	0	0	1	149
# dependent GitHub repos	5.26	0	0	0	0	677



(a) Repositories (b) Accounts

Figure 6: The Complementary Cumulative Distribution Function (CCDF) of activity durations for (a) repositories with fake star campaigns and sample repositories, (b) accounts with the lockstep signature and sample stargazers.

Prevalence in Package Registries. To identify the prevalence of repositories with fake star campaigns in existing package registries, we use the ecosyste.ms API [24] to query for explicit references to the GitHub repositories in our dataset.[3] We matched 738 packages (corresponding to 229 repositories); the most common registries are: PyPI (159 packages), npm (145), Pub (138), Cargo (123), and Go (118). We further collected their adoption statistics (i.e., # of dependent packages within the registry and # of dependent repositories in GitHub). Overall, results (Table 2) show scarce evidence of actual adoption for most of these packages despite their inflated star counts: 70.46%/77.50% of these packages do not even have a single dependent open-source package/repository, respectively.

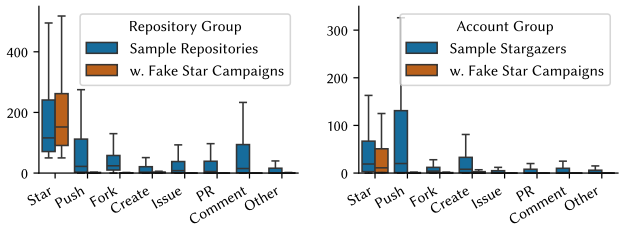
Finding 3: Only a small number (229; 1.23%) of repositories with fake star campaigns are also published in package registries (totaling 738 packages). Even fewer are widely adopted.

4.2 RQ2: Activity Patterns

Motivation. While StarScout detects anomalous starring patterns by construction (Section 3.2), it does not look into other types of GitHub activities. Understanding the latter may offer insights into how fake star campaigns operate and what they are used for. In this RQ, we examine the activity patterns of repositories and accounts with fake star campaigns from two different angles: (a) their overall duration of activity and (b) the distribution of activity types.

Duration of Activity. For repositories and accounts with fake star campaigns, we define their activity duration as the number of days from their first GHArchive event to their last. Since accounts with the low-activity signature (31.99% of accounts in fake star campaigns) have one day of activity by construction, we focus on the activity duration of accounts with the lockstep signature. For these latter accounts, we compare the Complementary Cumulative Distribution Function (CCDF) of their activity durations with the same comparison samples we used in Section 3.4 (Figure 6). We observe that most repositories with fake star campaigns are short-lived (Figure 6a): 83.90% of repositories with fake star campaigns have less than ten days of activity. However, user accounts with the lockstep signature can be active for long, with the overall distribution being similar to average GitHub stargazers (Figure 6b).

[3]This result is also an undercount, as explicit links to GitHub are optional.



(a) Repositories (b) Accounts

Figure 7: Comparing the distribution of GitHub events between: (1) repositories and accounts with fake star campaigns, and (2) random samples of repository and stargazers.

Finding 4: Most repositories with fake star campaigns are short-lived, but accounts can stay active for a long time.

Distribution of Activity Types. We aggregate the GHArchive events of repositories and accounts with fake star campaigns into eight types: *Star*, *Push*, *Fork*, *Create*, *Issue*, *PR* (i.e., pull requests), *Comment*, and *Other*. Then, we compare the overall distributions of activity types with the same samples we used in Section 3.4, aggregating the results in Figure 7. We can observe that repositories and accounts in fake star campaigns tend to be skewed toward starring activities, and they rarely engage in activities that are considerably harder to fake (e.g., issues, pull requests, comments).

However, some repositories and accounts may have different patterns compared to the trends in Figure 7. To explore this, we run the k-Means algorithm [27] under different ks on normalized activity vectors from repositories and accounts, respectively. We did not find meaningful clusters for repositories, but we find two additional clusters for accounts: Table 3 shows cluster centers for $k = 3$, which has the highest Silhouette Coefficient [76] among $k \in [2, 8]$. Apart from the majority of accounts that almost exclusively star repositories (Cluster 0, 77.75%), we find a cluster (Cluster 1, 14.76%) in which the accounts look generally more realistic—with pushes, repository creations, and other activities. Our manual analysis of a sample in Cluster 1 indicates that some of them may be creating and pushing artificial repositories to evade platform moderation, but there are also cases where we cannot tell whether they are real or not from their GitHub profiles. Finally, a small portion of accounts (Cluster 2, 7.49%) tend to engage in both starring and forking activities; we believe they may come from merchants selling both stars and forks.

Finding 5: Most repositories and accounts with fake star campaigns have trivial GitHub activity patterns.

Table 3: The properties of each clustering center identified by k-Means on accounts in fake star campaigns.

	% Star	% Push	% Fork	% Create	% Other
Cluster 0 (77.75%)	95.56%	1.00%	0.69%	2.10%	0.65%
Cluster 1 (14.76%)	28.55%	40.44%	2.55%	17.94%	10.52%
Cluster 2 (7.49%)	52.34%	1.08%	43.74%	1.76%	1.08%

Table 4: The most common words in the names of deleted repositories with fake star campaigns.

Category	Words and Occurrences
Pirated software	free (856), crack (721), pro (656), adobe (618), activation (467), cracked (263), studio (239), photoshop (177)
Cryptocurrency	bot (1,071), autoclicker (438), executor (321), crypto (312), wallet (241), trading (175), solana (154)
Game cheats	hack (357), cheat (256), roblox (252), h4ck (196)

Table 5: We categorize different groups of repositories with fake star campaigns (trending, packages, and others) into nine categories. Their frequencies are as follows.

Category	Trending	Packages	Other	All
Spam/Phishing	16 (20.5%)	22 (9.6%)	93 (31.1%)	131 (22.6%)
AI/LLM	25 (32.1%)	36 (15.7%)	45 (15.1%)	96 (16.6%)
Blockchain	10 (12.8%)	44 (19.2%)	30 (10.0%)	76 (13.1%)
Tool/Application	15 (19.2%)	22 (9.6%)	39 (13.0%)	73 (12.6%)
Tutorial/Demo	4 (5.1%)	6 (2.6%)	61 (20.4%)	70 (12.1%)
Web Frameworks	5 (6.4%)	44 (19.2%)	9 (3.0%)	56 (9.7%)
Basic Utility	0 (0.0%)	27 (11.8%)	3 (1.0%)	30 (5.2%)
Database	0 (0.0%)	10 (4.4%)	4 (1.3%)	14 (2.4%)
Other	3 (3.8%)	17 (7.4%)	15 (5.0%)	34 (5.9%)

4.3 RQ3: Repository Characteristics

Motivation. While RQ1 and RQ2 provide statistics on the detected repositories and accounts with fake star campaigns, they do not provide evidence regarding *what exactly these repositories are.* The anecdotes from the gray literature suggest that repositories may buy fake stars for popularity contests, growth hacking, or convincing investors [9, 13]; malicious actors may orchestrate fake star campaigns to spread malware phishing repositories [21]. Can we confirm these reports on StarScout's detections at a larger scale? What are the most common domains where fake star campaigns occur? We answer these subquestions in RQ3.

Analysis of Deleted Repositories. Our first analysis focuses on the 90.42% of repositories that have already been deleted on GitHub. Unfortunately, it is very difficult to study these repositories: They become inaccessible immediately after deletion from GitHub and GHArchive does not store any repository content.[4] Despite this, we can still glean some information from a term frequency analysis of their *repository names* (Table 4). We find that most of the deleted repositories carry names indicating pirated software (e.g., `Adobe-Animate-Crack`), cryptocurrency bots (e.g., `pixel-wallet-bot-free`, `Solana-Sniper-Bot`), or game cheats (e.g., `GTA5-cheat`). While GitHub may have taken some of them down due to copyright violations, it is probably not the majority case: The gray literature shows that fake star campaigns are used to promote phishing malware repositories [21]. Combining with the evidence we obtained from non-deleted repositories sharing similar names with deleted ones (e.g., Figure 1, more examples in Appendix, Section 6), we infer that most of these repositories could be phishing malware repositories masquerading as pirated software, game cheats, and cryptocurrency bots.

> **Finding 6:** Most (90.42%) repositories with fake star campaigns have been deleted on GitHub. Existing evidence suggests that they are probably phishing malware repositories masquerading as pirated software, game cheats, and cryptocurrency bots.

Analysis of Non-Deleted Repositories. For the remaining non-deleted repositories (1,783 in total), we clone them immediately at the time of identification. Then, we conduct open coding [58] on the repository READMEs—referring to other files where necessary—on three groups of repositories: (a) the 78 repositories that appeared in GitHub Trending (Section 4.1), which clearly reaches a wide range of developers; (b) the 229 repositories that are linked by open-source packages (Section 4.1), which may have different characteristics compared to others; (c) 299 randomly sampled repositories from the remaining 1,476 repositories (sample size determined from 95% confidence level and 5% margin of error). From this analysis, we identify eight major categories of repositories (Table 5). Among them, the most common—and also most concerning—category, is Spam/Phishing (31.1% in Other, 22.6% in All), where the repository is either an obvious clickbait (e.g., claiming game cheats, pirated software, cryptocurrency bots) to trick download of suspicious files or spreading spam content (e.g., used for search-engine-optimizations).[5] The remaining popular categories include: (1) AI/LLM, many of which are academic paper repositories or LLM-related startup products, (2) Blockchain or cryptocurrency related repositories, (3) Tool/Applications, or (4) Tutorials/Demos that seem to serve a reference or demonstration purpose (e.g., Python coding examples or list of awesome LLM tools). We provide definition of each category, analysis of inter-rater reliability, and examples of Spam/Phishing repositories in Appendix (Section 6).

> **Finding 7:** A significant portion (~30%) of repositories with fake star campaigns, that are still accessible on GitHub at the time of writing (i.e., March 2025), are spam or active phishing malware repositories. The remaining ones are mostly AI/LLM, blockchain, tool/applications, or tutorial/demo repositories.

4.4 RQ4: Promotional Effect

Motivation. For the first three research questions, we have presented a set of exploratory results around the prevalence of fake stars and the characteristics of repositories and accounts with fake

[4] World of Code [64] (WoC), another possible source of repository data, contains only a small portion of these deleted repositories—we believe they are likely unrepresentative of the remaining ones, depending on the specific discovery mechanisms used in WoC.

[5] Note that while we try our best to identify spam/phishing repositories, it is still possible that we miss better-obfuscated malicious repositories and mis-categorize them into other categories. We will discuss more on this limitation in Section 5.

star campaigns. While our RQ3 results highlight that the majority of repositories with fake star campaigns are likely short-lived phishing/spam repositories, there are still a non-negligible number of repositories in our dataset that remain online longer and seem more legitimate. Why would they buy fake GitHub stars? The gray literature often argues that (non-malicious) GitHub repositories buy fake stars for growth hacking [31] (i.e., "fake it until you make it"), especially because the number of stars is often considered a key indicator of the success of open-source projects and the occasional start-up companies associated with them [9]. Motivated by these speculations, we empirically analyze whether buying fake stars is effective in attracting substantial additional *real* attention, or only effective in making the repositories momentarily seem a little more popular (by the amount of stars they buy).

Hypotheses. We formulate the following two hypotheses regarding the effect of (fake and real) GitHub stars:

- **H_1:** *Accumulating real GitHub stars will help a GitHub repository gain more real GitHub stars in the future.*
- **H_2:** *Accumulating fake GitHub stars will help a GitHub repository gain more real GitHub stars in the future, but the effect is not as strong as that of real stars.*

We formulate $\mathbf{H}_1$ based on the well-known "rich get richer" phenomenon confirmed repeatedly in online social media and similar online contexts [51, 99]. We formulate $\mathbf{H}_2$ because there is no straightforward way for a GitHub user to distinguish fake stars from real ones while browsing a repository's web page (indeed, we had to build and run a tool for this), and thus fake stars should have a similar effect as real stars for most users. However, some users may look for other signals (e.g., issues and pull requests) and decide not to star the repository if other signals indicate poor quality, so we expect the effect to be smaller compared to real stars.

Methodology. To test our hypotheses, we fit *panel autoregression* models [52], estimating the longitudinal effect of gaining fake stars on attracting real stars in the future. Panel autoregression models are a family of regression models tailored to work on panel data (e.g., variables are collected per repository in a series of time periods) with temporal dependencies (i.e., variables at time t may be affected by variables at time $t - \Delta t$). By adding fixed effect or random effect terms to the model (we report on both specifications below, to demonstrate that our results are robust), we will be able to estimate the longitudinal effect of independent variables robustly to unobserved heterogeneity (i.e., factors that may affect the outcome variable but are not measured in the model). Specifically, we collect the following variables for each repository i in each month t:

- $fake_{i,t}$: Count of fake stars repository i gained in month t;
- $all\_fake_{i,t}$: Total count of fake stars in repository i as of t;
- $real_{i,t}$: Count of non-fake stars repository i gained in month t;
- $all\_real_{i,t}$: Total count of non-fake stars in repository i as of t;
- $age_{i,t}$: Number of months since repository i creation;
- $release_{i,t}$: Whether repository i has at least one GitHub release before or during month t;
- $activity_{i,t}$: The number of GHArchive events for repository i in month t, that come from GitHub users who are not the repository owner or one of the fake stargazers. This is a proxy measure for the amount of authentic development activities in month t.

Table 6: Fixed/random effect panel autoregression results.

	Dependent Variable: $real_{i,t}$	
	Random Effect	Fixed Effect
$real_{i,t-1}$	0.496^{***} (0.015)	0.364^{***} (0.018)
$real_{i,t-2}$	0.187^{***} (0.014)	0.148^{***} (0.015)
$all\_real_{i,t-3}$	0.069^{***} (0.008)	0.097^{***} (0.021)
$fake_{i,t-1}$	0.058^{***} (0.011)	0.074^{***} (0.012)
$fake_{i,t-2}$	0.024^{**} (0.010)	0.029^{**} (0.011)
$all\_fake_{i,t-3}$	-0.026^{***} (0.006)	-0.045^{***} (0.014)
$age_{i,t}$	-0.004^{***} (0.001)	
$release_{i,t}$	0.084^{***} (0.022)	
$activity_{i,t}$	0.087^{***} (0.010)	
Observations	12,738	12,738
R^2	0.665	0.268
Adjusted R^2	0.664	0.202
Note:	*p<0.1; **p<0.05; ***p<0.01	

The first four variables are intended to test our hypotheses; the remaining three are control variables that have been shown to correlate with star increases in prior work [48]. We fit $AR(k)$ (i.e., k-th order) models for $k = 1, 2, ..., 6$, defined as follows:

$$real_{i,t} \sim \sum_{j=1}^{k} real_{i,t-k} + all\_real_{i,t-k-1} + \sum_{j=1}^{k} fake_{i,t-k} + all\_fake_{i,t-k-1} + age_{i,t} + release_{i,t} + activity_{i,t}$$

The last three control variables are for random effect models only; unobserved heterogeneity in fixed effect models is controlled by the per-time and per-repository fixed-effect terms. All variables with skewed distributions are log-transformed before they are fed into the model. We fit the models using the R `plm` package [19].

Limitations. Before diving into the modeling results, we note the inherent limitations of our modeling approach. First, even if it allows for robust estimates of coefficients to represent the effect of independent variables, a regression model like ours may still provide insufficient evidence for causality claims. While Granger causality tests are often used in the econometric literature to establish stronger evidence of causality [44], the majority of time series in our data are too short to fulfill the minimum requirements of Granger causality tests on unbalanced panels ($t \geq 6 + 2k$, k is the autoregression order). More importantly, it is possible that real causality hides in exogenous variables [46] (e.g., if the repository maintainer is technically inexperienced and shortsighted, that may both cause the maintainer to buy fake stars and the repository to gain fewer real stars). Therefore, while our results provide initial evidence on the effect of fake stars on GitHub, future work is necessary to establish stronger evidence of causality and further uncover the mechanisms behind such effects.

Results. In this paper, we report results from the $AR(2)$ fixed effect and random effect model in Table 6. It is worth noting that we obtain consistent findings across all $AR(k)$ models; the remaining results and additional robustness checks are available in the paper Appendix (Section 6). We find strong support for $\mathbf{H}_1$: According to the fixed-effects model, a 1% increase of real stars in month

$t-1$ is correlated with an expected 0.36% increase of real stars in month t (since both variables are log-transformed in the model, the coefficient estimates correspond to percentage changes in the outcome) holding all other variables constant. Analogously, one can expect a 0.36% increase of real stars from month t to month $t+1$. The effect decreases to 0.15% in month $t+2$ and to 0.10% for all subsequent months, but the effect is always positive. In other words, a repository with more real stars tends to also get more real stars in the future, echoing the "rich get richer" phenomenon prevalent in social networks [51, 99].

On the other hand, $\mathbf{H}_2$ is only partially supported: while a 1% increase of fake stars in month t is associated with an expected 0.07% increase of real stars in month $t+1$ and 0.03% in month $t+2$, holding all other variables constant. In other words, fake stars do have a statistically significant, longitudinally decreasing positive effect on attracting real stars in the next two months, but the effect is about 5x smaller than that of real stars. However, a 1% increase of fake stars in month t is correlated with an expected 0.04% *decrease* (note the negative coefficient) of real stars on average for all months since month $t+2$. This suggests that buying fake stars may only help the repository gain new attention in the short term (i.e., less than two months), but the history of faking stars becomes a liability and may result in less overall attention in the long term, perhaps because the repository lacks in other activity and health indicators.

Finding 8: Buying fake stars may help a repository gain real attention in the short term (i.e., less than two months), but the effect is about 5x smaller compared to real stars; in the long term, buying fake stars has a negative effect on star gain.

5 Discussion: What to Do about Fake Stars?

Our study points to a growing problem: Fake star campaigns have become orders of magnitude more common in 2024 compared to 2023, further distorting the already questionable reliability of star count as a signal of popularity and trustworthiness. Most alarmingly, fake stars are demonstrably associated with increasing security risks and spam/phishing activities. In this final section, we discuss potential countermeasures to this problem from the lens of different stakeholders who may take the responsibility.

5.1 Open-Source Practitioners

Our study is far from the first showing that stars on GitHub are not a reliable signal [2, 55, 60, 77]. We provide further evidence that stars may not only misrepresent popularity or reputation, but can be intentionally manipulated by malicious actors. Yet, study after study shows how practitioners use star counts as an important (and often as the most important) signal for all kinds of decisions, including high-stakes decisions like open-source component adoption [66, 86, 92] and repository reputation evaluation [73, 85].

We suggest that *star counts should not be used for high-stakes decisions by themselves*. Yet, we see little hope for practitioners to change this widespread practice at scale any time soon, since it is already deeply ingrained and there are no immediately available and obvious alternatives. Still, we hope that exposing the security risks from such reliance may help motivate some changes until better, hopefully empirically validated, signals from platforms or third parties become established (Section 5.2 and 5.3).

In addition, while it is natural for open source developers to promote and try to attract attention to their projects [31, 47, 48], *we recommend against buying fake stars for growth hacking*. Not only is it arguably unethical and in violation of GitHub's terms of service, but also our RQ4 results suggest that it is ineffective. Our interpretation is that there is no plausible mechanism to convert stars into real, sustained adoption, even if a high star count may increase project visibility in the short term. Instead, we recommend that *repository maintainers and startup founders in open source should focus on strategies to actually build open-source communities, referring to the vast amount of literature on this topic* [e.g., 8, 59].

5.2 Platform Providers

Platform providers such as GitHub have a lot of leverage for designing features like stars and for designing security mechanisms against their abuse, and they have privileged information such as IP addresses from users that are not publicly available to researchers and third parties. We believe that *GitHub would benefit from a better designed popularity signal other than raw star counts*. The current system, which gives equal weight to all stars, is prone to manipulation from bots and coordinated inauthentic activities. The literature on review and reputation systems has many ideas for designing more robust measures that can provide inspiration, such as (1) highlighting signals of real adoption as a popularity measure (e.g., based on dimensions of network centrality [55]); (2) incorporating a reputation system where stars of more active or authentic users or of developers actually adopting the dependency are weighted higher [e.g., 42, 65]), and (3) shadowbanning stars from suspicious users [67, 75]. Stack Overflow's reputation system and Yelp's review filtering are examples of deliberate designs to combat low quality and malicious user inputs. Our StarScout tool is immediately actionable for a shadowbanning approach.

Another key takeaway from our research findings is that *fake stars are highly related to malicious activities and should be considered as such in platform moderation*. Our manual investigation of repositories in RQ3 reveals multiple cases of malicious repositories that had been stripped of all their stars but still remained accessible on GitHub (examples in Appendix, Section 6). We speculate that GitHub may routinely take down known fake accounts or malware repositories, but may not link them together. We suggest that a platform provider could use internal signals of suspicious activity, including fake stars, to direct malware detection efforts.

Finally, our study provides an example of *how platform transparency enables large-scale independent studies of fraud and malicious activities*. We could not have conducted this study without GitHub's high level of openness, which unfortunately, is not common in many other online platforms. *We call for the same level of openness in others to enable "the scrutiny of a thousand eyes."*

5.3 Third-Party Tool Providers

While the platform itself has the most direct access to affect change for all its users, third parties can also offer valuable services for the entire community. They may also be more frank in calling out suspicious practices than a platform owner may be comfortable.

In our case, *software composition analysis (SCA) tool providers are a good match for auditing suspicious activities in the dependencies that they are monitoring for their customers.* For example, they may flag dependencies with suspicious fake stars to encourage developers to review their adoption decision and any unusual behaviors or security risks inside the repository. To demonstrate the feasibility of suspicious star detection through a third-party tool, we have collaborated with Socket Inc (a) to inform their customers about the threat from fake stars and (b) to integrate our detector to produce alerts for their customers if any of their dependencies have a history of fake star campaigns. The rationale for this intervention is that the adoption of an open-source component with fake stars deserves stakeholder attention, and we expect this alert to be used with other supply chain security alerts implemented in the company's product for stakeholder re-evaluation. At the time of writing, the company's blog post about fake stars had over 2.4k page views, and their tool generated alerts for 283 out of two million distinct customer Software Bill of Materials (SBOMs). As expected, this alert currently only applies to a small percentage of SBOMs in industry software projects, as we only identified 229 repositories with fake star campaigns and also published in package registries (Section 4.1). Yet, the fact that these packages are found in actual customer SBOMs underscores the practical relevance of our research findings and signals the potential for mitigation through third-party tools.

Third-party tools can also provide better popularity and trustworthiness signals to replace (or at least supplement) star counts in decision-making. For example, there have been a number of ongoing projects that identify security-relevant and activity-based signals for open-source component adoption [e.g., 98].

5.4 Researchers

With our study, we call for further research on spam and fraudulent activities in the software supply chain. While spam and fraud are relatively well studied in emails [30], social media [26], and e-commerce [54], spam/fraud research on the software supply chain has been scarce (with only a few exceptions [43, 94]). However, both our findings, and recent gray literature reports [5, 15, 16] indicate that spam and fraudulent activities are rapidly rising in platforms supporting the current software supply chain, including GitHub, npm, PyPI, and DockerHub. This arguably creates an emerging and important battleground, especially considering their wide usage and critical importance to our modern digital infrastructure [45]. Also, software supply chain attacks are becoming increasingly diverse and sophisticated, evolving from simple typosquatting [69] to complex social engineering attacks as in the *xz* attack [11] and reputation farming [18]. *Although our study does not find any evidence of fake stars being used for social engineering attacks, we call for continuous research and monitoring into this potential threat.*

6 Conclusion

In this paper, we have presented a systematic, global, and longitudinal measurement study of fake stars in GitHub. Our study sheds light on this prevalent and escalating threat happening in a platform central to modern open-source software development. Despite its wide prevalence and rising popularity, our findings have also revealed its shady nature: Fake stars are, at worst, used to spread malware in short-lived repositories, and at best, used as a short-term promotional tool, which does not bring long-term returns. Our work emphasizes a critical need for vigilance in assessing other repository signals beyond star counts, and calls for the design of better popularity signals. Our study also shows a need for further research on spam, fraudulent, and malicious activities in the software supply chain, especially regarding the risk of social engineering attacks. Future research could expand upon our findings through better fake star detection approaches, exploring additional data (e.g., analyzing fake accounts in GitHub from a social network perspective), and examining similar spam, fraudulent, or malicious activities in other platforms related to software supply chain security.

Responsible Disclosure

For all Spam/Phishing repositories we find in RQ3 that were still accessible on GitHub when we discovered them, we reported them immediately to GitHub. Based on their phishing patterns, we further identified all remaining repositories in our dataset that (1) provide a download link to `.zip`, `.rar` or `.exe` file, (2) VirusTotal [23] reports the presence of malware, or (3) the README content is highly similar to other repositories marked as Spam/Phishing by our standard. We manually verified the identified repositories and reported those that align with our open coding criteria for Spam/Phishing repositories in RQ3. In total, we reported 130 phishing repositories to GitHub, plus two accounts that appear to be almost entirely used for Search Engine Optimization (SEO) spamming (each of them having 213 and 139 repositories, respectively). At the time of writing (August 2025), all phishing repositories and one of the SEO spamming accounts are no longer accessible in GitHub; the other SEO spamming account only has 17 public repositories remaining. For the remaining repositories and accounts with suspected fake stars, we do not directly report them, given the possibility that StarScout may generate false positives and thus false accusations (recall the discussion in Section 3.5). Still, we reached out to the GitHub Security team to inform them of our research—they likely have access to other non-public information, such as IP addresses, that could be used to further validate and refine our detections.

Data Availability

We provide the appendix of this paper, the source code of StarScout, and the data and scripts used in our measurement study at:

https://doi.org/10.5281/zenodo.17009693

The appendix is also available in the arXiv version of this paper [53].

Acknowledgments

He's and Kästner's work was supported in part by the National Science Foundation (award 2206859). Kapravelos's work was supported in part by the National Science Foundation (award 2207008). Bogdan's work was supported in part by the National Science Foundation (award 2317168) and in part by research awards from Google and the Digital Infrastructure Fund. We would also like to thank Socket Inc (https://socket.dev) and its CEO, Feross Aboukhadijeh, for providing collaboration opportunities, general support, and computational resources for this project. Additionally, we would like to thank Google Cloud for Researchers program for offering credits to fund the use of Google BigQuery for this research.

References

[1] 2011. *GHArchive.* Retrieved Sept 19, 2024 from https://www.gharchive.org/

[2] 2019. *Can we trust GitHub stars?* Retrieved July 9, 2024 from https://traefik.io/blog/can-we-trust-github-stars-e8aa8b6b0baa/

[3] 2021. *Controversy arises as Ali OceanBase offers gifts for GitHub stars, CTO apologizes.* Retrieved July 9, 2024 from https://mp.weixin.qq.com/s/q62sSpd6y7_JGt7mKG0LbQ

[4] 2022. *GitHub 'Patriotic Package,' Actually Just Selling Stars.* Retrieved Sept 24, 2024 from https://juejin.cn/post/7115303814483148807

[5] 2022. *Phishing Campaign Targets PyPI Users to Distribute Malicious Code.* Retrieved Nov 14, 2024 from https://www.darkreading.com/cloud-security/phishing-campaign-targets-pypi-users-to-distribute-malicious-code

[6] 2023. *Malicious Microsoft VSCode extensions steal passwords, open remote shells.* Retrieved Sep 23, 2024 from https://www.bleepingcomputer.com/news/security/malicious-microsoft-vscode-extensions-steal-passwords-open-remote-shells/

[7] 2023. *NPM Account Takeover Results in Crypto Supply Chain Attack.* Retrieved Apr 28, 2024 from https://checkmarx.com/blog/npm-account-takeover-results-in-crypto-supply-chain-attack/

[8] 2023. *The product approach to open-source communities.* Retrieved Mar 11, 2025 from https://stackoverflow.blog/2023/11/08/the-product-approach-to-open-source-communities/

[9] 2023. *Should startups worry about GitHub stars?* Retrieved July 9, 2024 from https://technical.ly/software-development/should-startups-worry-about-github-stars/

[10] 2023. Tracking the Fake GitHub Star Black Market with Dagster, dbt and BigQuery. Retrieved July 9, 2024 from https://dagster.io/blog/fake-stars

[11] 2024. *Everything you need to know about the xz util backdoor.* Retrieved Sept 18, 2024 from https://www.wired.com/story/xz-backdoor-everything-you-need-to-know/

[12] 2024. *GitHub Acceptable Use Policies.* Retrieved Sep 24, 2024 from https://docs.github.com/en/site-policy/acceptable-use-policies

[13] 2024. *The GitHub Black Market That Helps Coders Cheat the Popularity Contest.* Retrieved July 9, 2024 from https://www.wired.com/story/github-stars-black-market-coders-cheat/

[14] 2024. *GitStar.* Retrieved Sept 24, 2024 from https://gitstar.com.cn/

[15] 2024. *The Great npm Garbage Patch.* Retrieved Nov 11, 2024 from https://blog.phylum.io/the-great-npm-garbage-patch/

[16] 2024. *Millions of Malicious Containers Found on Docker Hub.* Retrieved Nov 14, 2024 from https://www.infosecurity-magazine.com/news/malicious-containers-found-docker/

[17] 2024. *Open Source Supply, Demand, and Security.* Retrieved Sept 18, 2024 from https://www.sonatype.com/state-of-the-software-supply-chain/open-source-supply-and-demand

[18] 2024. *OpenSSF Warns of Reputation Farming Leveraging Closed GitHub Issues and PRs.* Retrieved Sept 23, 2024 from https://socket.dev/blog/openssf-warns-of-reputation-farming-using-closed-github-issues-and-prs

[19] 2024. *plm: Linear Models for Panel Data.* Retrieved Apr 28, 2024 from https://cran.r-project.org/web/packages/plm

[20] 2024. *Rate limits for the REST API.* Retrieved Sept 18, 2024 from https://docs.github.com/en/rest/using-the-rest-api/rate-limits-for-the-rest-api

[21] 2024. *Stargazer Ghost Network.* Retrieved Sept 17, 2024 from https://research.checkpoint.com/2024/stargazers-ghost-network/

[22] 2024. *Trending repositories on GitHub today.* Retrieved Oct 28, 2024 from https://github.com/trending

[23] 2024. *VirusTotal.* Retrieved Nov 8, 2024 from https://www.virustotal.com/

[24] 2025. *ecosyste.ms.* Retrieved Feb 25, 2025 from https://ecosyste.ms/

[25] 2025. *larsbijl/trending_archive.* Retrieved Feb 25, 2025 from https://github.com/larsbijl/trending_archive

[26] Malak Saleh Aljabri, Rachid Zagrouba, Afrah Shaahid, Fatima Alnasser, Asalah Saleh, and Dorieh M. Alomari. 2023. Machine learning-based social media bot detection: A comprehensive literature review. *Soc. Netw. Anal. Min.* 13, 1 (2023), 20.

[27] David Arthur and Sergei Vassilvitskii. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007.* SIAM, 1027–1035.

[28] Andrew Begel, Jan Bosch, and Margaret-Anne D. Storey. 2013. Social Networking Meets Software Development: Perspectives from GitHub, MSDN, Stack Exchange, and TopCoder. *IEEE Softw.* 30, 1 (2013), 52–66.

[29] Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. 2013. CopyCatch: Stopping group attacks by spotting lockstep behavior in social networks. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013.* ACM, 119–130.

[30] Enrico Blanzieri and Anton Bryl. 2008. A survey of learning-based techniques of email spam filtering. *Artif. Intell. Rev.* 29, 1 (2008), 63–92.

[31] René Bohnsack and Meike Malena Liesner. 2019. What the hack? A growth hacking taxonomy and practical applications for firms. *Business Horizons* 62, 6 (2019), 799–818.

[32] Hudson Borges and Marco Túlio Valente. 2018. What's in a GitHub Star? Understanding Repository Starring Practices in a Social Coding Platform. *J. Syst. Softw.* 146 (2018), 112–129.

[33] Donald T Campbell. 1979. Assessing the impact of planned social change. *Evaluation and Program Planning* 2, 1 (1979), 67–90.

[34] Alan Cao and Brendan Dolan-Gavitt. 2022. What the fork? Finding and analyzing malware in GitHub forks. In *Proc. of NDSS*, Vol. 22.

[35] Qiang Cao, Xiaowei Yang, Jieqi Yu, and Christopher Palow. 2014. Uncovering Large Groups of Active Malicious Accounts in Online Social Networks. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014.* ACM, 477–488.

[36] David Nevado Catalán, Sergio Pastrana, Narseo Vallina-Rodriguez, and Juan Tapiador. 2023. An analysis of fake social media engagement services. *Comput. Secur.* 124 (2023), 103013.

[37] Ronald Jay Cohen, Mark E Swerdlik, and Suzanne M Phillips. 1996. *Psychological Testing and Assessment: An Introduction to Tests and Measurement.* Mayfield Publishing Co.

[38] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2015. Fame for sale: Efficient detection of fake Twitter followers. *Decis. Support Syst.* 80 (2015), 56–71.

[39] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2017. The Paradigm-Shift of Social Spambots: Evidence, Theories, and Tools for the Arms Race. In *Proceedings of the 26th International Conference on World Wide Web Companion.* ACM, 963–972.

[40] Emiliano De Cristofaro, Arik Friedman, Guillaume Jourjon, Mohamed Ali Kâafar, and Muhammad Zubair Shafiq. 2014. Paying for Likes?: Understanding Facebook Like Fraud Using Honeypots. In *Proceedings of the 2014 Internet Measurement Conference, IMC 2014.* ACM, 129–136.

[41] Alexandre Decan, Tom Mens, and Philippe Grosjean. 2019. An empirical comparison of dependency network evolution in seven software packaging ecosystems. *Empir. Softw. Eng.* 24, 1 (2019), 381–416.

[42] Chrysanthos Dellarocas. 2010. Online reputation systems: How to design one that does what you need. *MIT Sloan Management Review* 51, 3 (2010), 33.

[43] Kun Du, Hao Yang, Yubao Zhang, Haixin Duan, Haining Wang, Shuang Hao, Zhou Li, and Min Yang. 2020. Understanding Promotion-as-a-Service on GitHub. In *ACSAC '20: Annual Computer Security Applications Conference, Virtual Event / Austin, TX, USA, 7-11 December, 2020.* ACM, 597–610.

[44] Elena-Ivona Dumitrescu and Christophe Hurlin. 2012. Testing for Granger non-causality in heterogeneous panels. *Economic Modelling* 29, 4 (2012), 1450–1460.

[45] Nadia Eghbal. 2016. *Roads and Bridges: The Unseen Labor Behind Our Digital Infrastructure.* Ford Foundation.

[46] Robert F Engle, David F Hendry, and Jean-Francois Richard. 1983. Exogeneity. *Econometrica: Journal of the Econometric Society* (1983), 277–304.

[47] Hongbo Fang, Daniel Klug, Hemank Lamba, James D. Herbsleb, and Bogdan Vasilescu. 2020. Need for Tweet: How Open Source Developers Talk About Their GitHub Work on Twitter. In *MSR '20: 17th International Conference on Mining Software Repositories.* ACM, 322–326.

[48] Hongbo Fang, Hemank Lamba, James D. Herbsleb, and Bogdan Vasilescu. 2022. "This Is Damn Slick!" Estimating the Impact of Tweets on Open Source Project Popularity and New Contributors. In *44th IEEE/ACM 44th International Conference on Software Engineering, ICSE 2022.* ACM, 2116–2129.

[49] Robert Faris, Hal Roberts, Bruce Etling, Nikki Bourassa, Ethan Zuckerman, and Yochai Benkler. 2017. Partisanship, propaganda, and disinformation: Online media and the 2016 US presidential election. *Berkman Klein Center Research Publication* 6 (2017).

[50] Chris Grier, Kurt Thomas, Vern Paxson, and Chao Michael Zhang. 2010. @spam: The underground on 140 characters or less. In *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010.* ACM, 27–37.

[51] Nick Hagar and Aaron Shaw. 2022. Concentration without cumulative advantage: The distribution of news source attention in online communities. *Journal of Communication* 72, 6 (2022), 675–686.

[52] Christoph Hanck, Martin Arnold, Alexander Gerber, and Martin Schmelzer. 2021. Regression with Panel Data. In *Introduction to Econometrics with R.* Universität Duisburg-Essen. https://www.econometrics-with-r.org/10-rwpd.html

[53] Hao He, Haoqin Yang, Philipp Burckhardt, Alexandros Kapravelos, Bogdan Vasilescu, and Christian Kästner. 2024. Six Million (Suspected) Fake Stars in GitHub: A Growing Spiral of Popularity Contests, Spams, and Malware. *CoRR* abs/2412.13459 (2024). arXiv:2412.13459

[54] Li He, Xianzhi Wang, Hongxu Chen, and Guandong Xu. 2022. Online Spam Review Detection: A Survey of Literature. *Hum. Centric Intell. Syst.* 2, 1-2 (2022), 14–30.

[55] Runzhi He, Hengzhi Ye, and Minghui Zhou. 2024. Revealing the Value of Repository Centrality in Lifespan Prediction of Open Source Software Projects. *CoRR* abs/2405.07508 (2024). arXiv:2405.07508

[56] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. FRAUDAR: Bounding Graph Fraud in the Face of Camouflage. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge*

Discovery and Data Mining. ACM, 895–904.
[57] Muhammad Ikram, Lucky Onwuzurike, Shehroze Farooqi, Emiliano De Cristofaro, Arik Friedman, Guillaume Jourjon, Mohamed Ali Kâafar, and M. Zubair Shafiq. 2017. Measuring, Characterizing, and Detecting Facebook Like Farms. *ACM Trans. Priv. Secur.* 20, 4 (2017), 13:1–13:28.
[58] Shahedul Huq Khandkar. 2009. Open coding. *University of Calgary* (2009).
[59] Amy Jo Kim. 2006. *Community Building on the Web: Secret Strategies for Successful Online Communities*. Peachpit Press.
[60] Simon Koch, David Klein, and Martin Johns. 2024. The Fault in Our Stars: An Analysis of GitHub Stars as an Importance Metric for Web Source Code. In *MADWeb: Workshop on Measurements; Attacks and Defenses for the Web*.
[61] Tadayoshi Kohno, Yasemin Acar, and Wulf Loh. 2023. Ethical Frameworks and Computer Security Trolley Problems: Foundations for Conversations. In *32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, August 9-11, 2023*. USENIX Association, 5145–5162.
[62] Piergiorgio Ladisa, Henrik Plate, Matias Martinez, and Olivier Barais. 2023. SoK: Taxonomy of Attacks on Open-Source Software Supply Chains. In *44th IEEE Symposium on Security and Privacy, SP 2023*. IEEE, 1509–1526.
[63] Majd Latah. 2020. Detection of malicious social bots: A survey and a refined taxonomy. *Expert Syst. Appl.* 151 (2020), 113383.
[64] Yuxing Ma, Tapajit Dey, Chris Bogart, Sadika Amreen, Marat Valiev, Adam Tutko, David Kennard, Russell Zaretzki, and Audris Mockus. 2021. World of Code: Enabling a research workflow for mining and analyzing the universe of open source VCS data. *Empir. Softw. Eng.* 26, 2 (2021), 22.
[65] Dana Movshovitz-Attias, Yair Movshovitz-Attias, Peter Steenkiste, and Christos Faloutsos. 2013. Analysis of the reputation system and user contributions on a question answering website: StackOverflow. In *Advances in Social Networks Analysis and Mining 2013, ASONAM '13*. ACM, 886–893.
[66] Suhaib Mujahid, Rabe Abdalkareem, and Emad Shihab. 2023. What are the characteristics of highly-selected packages? A case study on the npm ecosystem. *J. Syst. Softw.* 198 (2023), 111588.
[67] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie Glance. 2013. What Yelp fake review filter might be doing?. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 7. 409–418.
[68] Nuthan Munaiah, Steven Kroh, Craig Cabrey, and Meiyappan Nagappan. 2017. Curating GitHub for engineered software projects. *Empir. Softw. Eng.* 22, 6 (2017), 3219–3253.
[69] Marc Ohm, Henrik Plate, Arnold Sykosch, and Michael Meier. 2020. Backstabber's Knife Collection: A Review of Open Source Software Supply Chain Attacks. In *Detection of Intrusions and Malware, and Vulnerability Assessment - 17th International Conference, DIMVA 2020, Lisbon, Portugal, June 24-26, 2020, Proceedings (Lecture Notes in Computer Science, Vol. 12223)*. Springer, 23–43.
[70] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. 2007. Netprobe: A fast and scalable system for fraud detection in online auction networks. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. ACM, 201–210.
[71] René Peeters. 2003. The maximum edge biclique problem is NP-complete. *Discret. Appl. Math.* 131, 3 (2003), 651–654.
[72] Francesco Pierri, Alessandro Artoni, and Stefano Ceri. 2020. Investigating Italian disinformation spreading on Twitter in the context of 2019 European elections. *PloS One* 15, 1 (2020), e0227821.
[73] Huilian Sophie Qiu, Yucen Lily Li, Hema Susmita Padala, Anita Sarma, and Bogdan Vasilescu. 2019. The Signals that Potential Contributors Look for When Choosing Open-source Projects. *Proc. ACM Hum. Comput. Interact.* 3, CSCW (2019), 122:1–122:29.
[74] Sanjeev Rao, Anil Kumar Verma, and Tarunpreet Bhatia. 2021. A review on social spam detection: Challenges, open issues, and future directions. *Expert Syst. Appl.* 186 (2021), 115742.
[75] Marten Risius and Kevin Marc Blasiak. 2024. Shadowbanning: An Opaque Form of Content Moderation. *Business & Information Systems Engineering* (2024), 1–13.
[76] Peter J Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* 20 (1987), 53–65.
[77] William Schueller and Johannes Wachs. 2024. Modeling interconnected social and technical risks in open source software ecosystems. *Collective Intelligence* 3, 1 (2024), 26339137241231912.
[78] Indira Sen, Anupama Aggarwal, Shiven Mian, Siddharth Singh, Ponnurangam Kumaraguru, and Anwitaman Datta. 2018. Worth its Weight in Likes: Towards Detecting Fake Likes on Instagram. In *Proceedings of the 10th ACM Conference on Web Science, WebSci 2018*. ACM, 205–209.
[79] Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. 2018. The spread of low-credibility content by social bots. *Nature Communications* 9, 1 (2018), 1–9.
[80] Jonghyuk Song, Sangho Lee, and Jong Kim. 2015. CrowdTarget: Target-based Detection of Crowdturfing in Online Social Networks. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*. ACM, 793–804.
[81] Gianluca Stringhini, Gang Wang, Manuel Egele, Christopher Kruegel, Giovanni Vigna, Haitao Zheng, and Ben Y. Zhao. 2013. Follow the green: Growth and dynamics in Twitter follower markets. In *Proceedings of the 2013 Internet Measurement Conference, IMC 2013*. ACM, 163–176.
[82] Nishat Ara Tania, Md Rayhanul Masud, Md Omar Faruk Rokon, Qian Zhang, and Michalis Faloutsos. 2024. Who is Creating Malware Repositories on GitHub and Why?. In *Companion Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, Singapore, May 13-17, 2024*. ACM, 955–958.
[83] Kurt Thomas, Chris Grier, Dawn Song, and Vern Paxson. 2011. Suspended accounts in retrospect: An analysis of Twitter spam. In *Proceedings of the 11th ACM SIGCOMM Internet Measurement Conference, IMC '11, Berlin, Germany, November 2-, 2011*. ACM, 243–258.
[84] Asher Trockman, Shurui Zhou, Christian Kästner, and Bogdan Vasilescu. 2018. Adding sparkle to social coding: An empirical study of repository badges in the *npm* ecosystem. In *Proceedings of the 40th International Conference on Software Engineering, ICSE 2018*. ACM, 511–522.
[85] Jason Tsay, Laura Dabbish, and James D. Herbsleb. 2014. Influence of social and technical factors for evaluating contribution in GitHub. In *36th International Conference on Software Engineering, ICSE '14*. ACM, 356–366.
[86] Enrique Larios Vargas, Maurício Finavaro Aniche, Christoph Treude, Magiel Bruntink, and Georgios Gousios. 2020. Selecting third-party libraries: The practitioners' perspective. In *ESEC/FSE '20: 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual Event, USA, November 8-13, 2020*. ACM, 245–256.
[87] Bimal Viswanath, Muhammad Ahmad Bashir, Mark Crovella, Saikat Guha, Krishna P. Gummadi, Balachander Krishnamurthy, and Alan Mislove. 2014. Towards Detecting Anomalous User Behavior in Online Social Networks. In *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014*. USENIX Association, 223–238.
[88] Duc-Ly Vu, Ivan Pashchenko, Fabio Massacci, Henrik Plate, and Antonino Sabetta. 2020. Typosquatting and Combosquatting Attacks on the Python Ecosystem. In *IEEE European Symposium on Security and Privacy Workshops, EuroS&P Workshops 2020, Genoa, Italy, September 7-11, 2020*. IEEE, 509–514.
[89] Gang Wang, Tianyi Wang, Haitao Zheng, and Ben Y. Zhao. 2014. Man vs. Machine: Practical Adversarial Detection of Malicious Crowdsourcing Workers. In *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014*. USENIX Association, 239–254.
[90] Gang Wang, Christo Wilson, Xiaohan Zhao, Yibo Zhu, Manish Mohanlal, Haitao Zheng, and Ben Y. Zhao. 2012. Serf and turf: Crowdturfing for fun and profit. In *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*. ACM, 679–688.
[91] Edgar R. Weippl, Sebastian Schrittwieser, and Sylvi Rennert. 2016. Empirical Research and Research Ethics in Information Security. In *Information Systems Security and Privacy - Second International Conference, ICISSP 2016, Rome, Italy, February 19-21, 2016, Revised Selected Papers (Communications in Computer and Information Science, Vol. 691)*. Springer, 14–22.
[92] Dominik Wermke, Noah Wöhler, Jan H. Klemmer, Marcel Fourné, Yasemin Acar, and Sascha Fahl. 2022. Committed to Trust: A Qualitative Study on Security & Trust in Open Source Software Projects. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*. IEEE, 1880–1896.
[93] Evan D Wolff, KatE M Growley, Maida O Lerner, Matthew B Welling, Michael G Gruden, and Jacob Canter. 2021. Navigating the SolarWinds supply chain attack. *Procurement Law.* 56 (2021), 3.
[94] Mengying Wu, Geng Hong, Wuyuao Mai, Xinyi Wu, Lei Zhang annd Yingyuan Pu, Huajun Chai, Lingyun Ying, Haixin Duan, and Min Yang. 2025. Exposing the Hidden Layer: Software Repositories in the Service of SEO Manipulation. In *ICSE 2025: The 47th International Confenrece on Software Engineering*.
[95] Cao Xiao, David Mandell Freeman, and Theodore Hwa. 2015. Detecting Clusters of Fake Accounts in Online Social Networks. In *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security, AISec 2015*. ACM, 91–101.
[96] Kai-Cheng Yang, Onur Varol, Clayton A Davis, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2019. Arming the public with artificial intelligence to counter social bots. *Human Behavior and Emerging Technologies* 1, 1 (2019), 48–61.
[97] Zhi Yang, Christo Wilson, Xiao Wang, Tingting Gao, Ben Y. Zhao, and Yafei Dai. 2014. Uncovering social network Sybils in the wild. *ACM Trans. Knowl. Discov. Data* 8, 1 (2014), 2:1–2:29.
[98] Nusrat Zahan, Parth Kanakiya, Brian Hambleton, Shohanuzzaman Shohan, and Laurie A. Williams. 2023. OpenSSF Scorecard: On the Path Toward Ecosystem-Wide Automated Security Metrics. *IEEE Secur. Priv.* 21, 6 (2023), 76–88.
[99] Linhong Zhu and Kristina Lerman. 2016. Attention Inequality in Social Media. *CoRR* abs/1601.07200 (2016). arXiv:1601.07200
[100] Markus Zimmermann, Cristian-Alexandru Staicu, Cam Tenny, and Michael Pradel. 2019. Small World with High Risks: A Study of Security Threats in the npm Ecosystem. In *28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August 14-16, 2019*. USENIX Association, 995–1010.