GitHub Abuse Flaw Shows Why We Can’t Shrug Off Abuse Vulnerabilities in Security

By Dvir Sasson — Director of Security Research at Reco AI May 13, 2024

Published: 2024-05-13 · Archived: 2026-04-05 21:51:48 UTC

Security has always been a game of risk management, not risk elimination. Every decision to address one threat means potentially leaving another unattended. That deciding of which threat to address – and in what order – is the name of the game. In this triage process, abuse vulnerabilities, *i.e.*, exploiting legitimate features of a platform in unintended ways to conduct digital misdeeds such as phishing campaigns, can get pushed down the priority list of security issues. I would like to argue that it's time we stop separating the concept of abuse vulnerabilities and security vulnerabilities.

Unlike security vulnerabilities that are, in essence, exploited loopholes or bugs in the code, fixes for abuse vulnerabilities can be slow to come. Yet these openings for abuse can easily lead to disaster if left unattended. Recent figures show that 68% of breaches originate from these exact types of exploitations involving the human element making a mistake such as phishing attempts or abuse vulnerabilities.

As an example, let's examine a recent abuse vulnerability I found in GitHub. This vulnerability bypasses traditional email security controls and takes advantage of GitHub's lack of content sanitization, which enables the distribution of malicious content on its platform.

Refresher: What is GitHub?#

GitHub is a widely popular code hosting platform owned by Microsoft. It allows developers to collaborate, share code, and manage projects effectively. With its widespread adoption and integration into modern development workflows, GitHub has become a crucial tool for many organizations, especially in software development and production. In turn, this widespread adoption has led to threat actors increasingly targeting it in order to gain access to proprietary code, keys/access to systems, and more.

A Summary of GitHub's Abuse Vulnerability#

While GitHub offers powerful collaboration features, we discovered a way to abuse its functionality and bypass security configurations to target organizational users with spear-phishing emails. Here are some of the highlights of this issue.

1. **Mention Mechanism:** When attempting to mention a user with an "@" sign, GitHub suggests usernames. However, if you know a specific user's username (even within an organization), mentioning them will notify them via their configured notification method, which is typically email by default. This gives threat actors the ability to use GitHub's notification as a way to email and contact users in an organization that *will not be caught by email security mechanisms*.
2. **Lack of Privacy Controls:** GitHub does not allow users to set granular privacy controls for who can mention or tag them. This lack of control means that anyone can potentially mention a user, even if they are not part of the same organization or project.
3. **Global Notification Settings:** GitHub only allows users to set their notification preferences globally, without fine-grained policies or whitelisting mechanisms.
4. **Abuse Persistence:** While GitHub allows users to block and report threat actors and their issues, the attacker can simply create a new user account, repository, and issue to restart the attack. This is made infinitely easier with AI and

other bots that make this process trivial for threat actors.

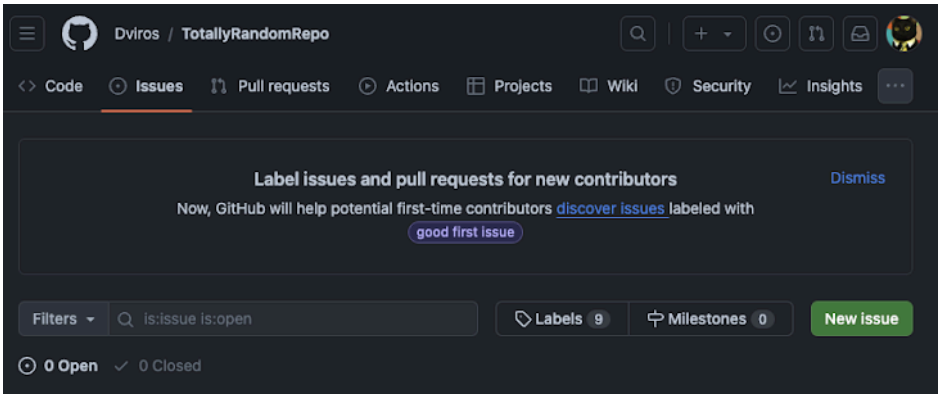
5. **No Content Validation:** GitHub does not perform any syntax validation or leverage language models to review content, potentially allowing malicious payloads to slip through.

The Attack Kill Chain#

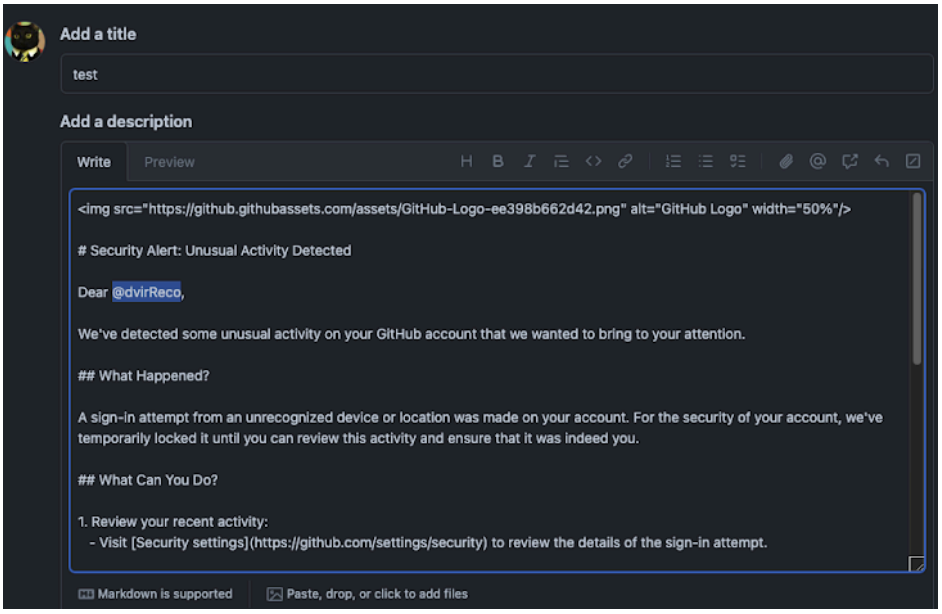
Here's a step-by-step breakdown of how threat actors can exploit this vulnerability:

1. **Set up Repository and Issue:** The attacker creates a new repository or piggybacks on an existing one, then creates a new issue within it.

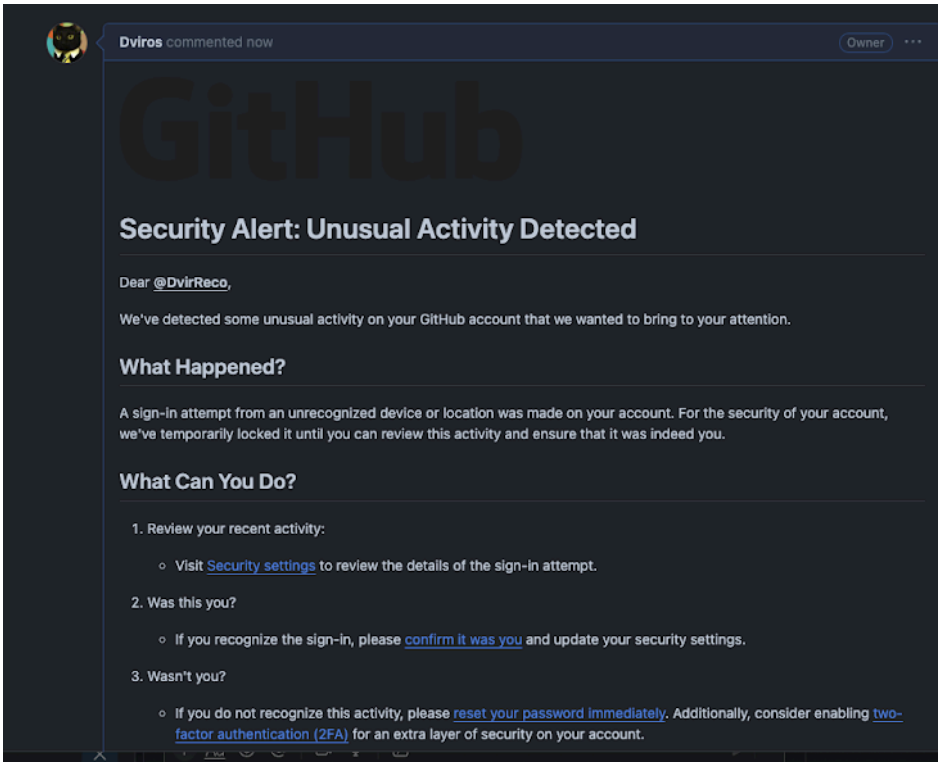
Threat actors spin up a new repository, or even leeching on existing ones, creating new issues.



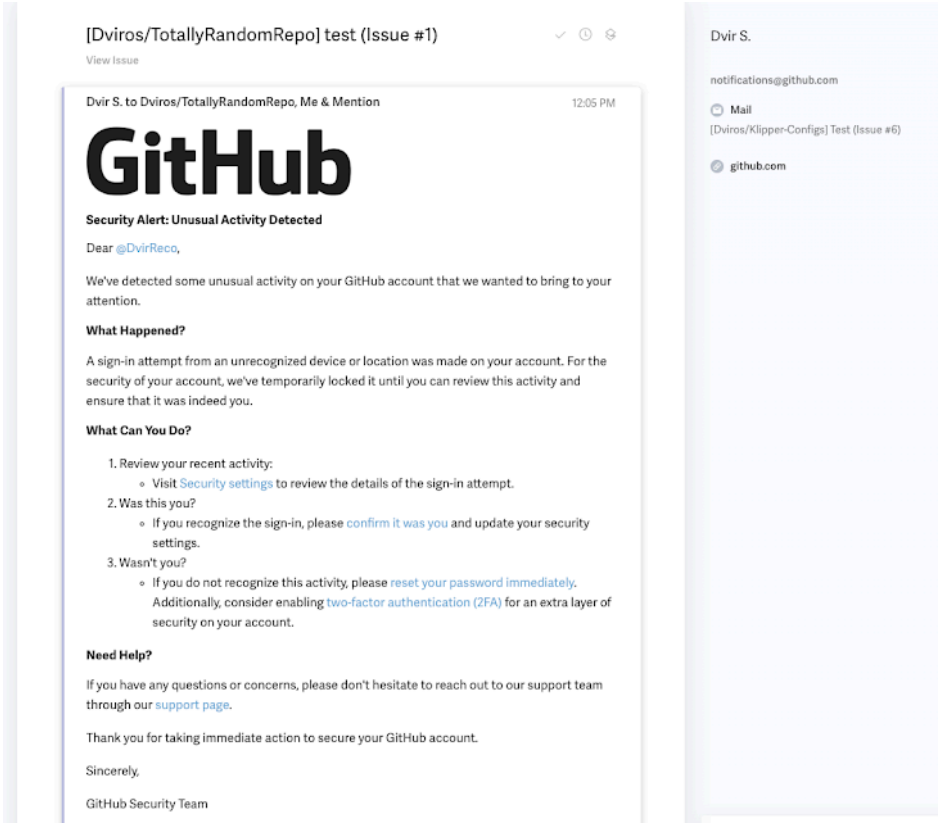
Within the issue, threat actors are able to create a Markdown based formatting. We even used an LLM to generate the following text as an example of a threat actor creating an official-looking lure asking the user to sign into their account and change their password.



By replacing the username with the correct victim's username, we can make sure that the email will be delivered safely, bypassing email security measures, because these email security mechanisms will assume that an email "from" GitHub alerting of a notification, similar to other such notifications necessary for the developer to perform their jobs.



Once submitted, the victim will receive the phishing email lure.



As you can see, the display name of the sender is derived from the abusing user's display name. The threat actor could easily even change this to something such as "GitHub Security" by changing the display name and not the user name.

The malicious sender is notifications@github.com, which means it's a reputable sender that's being abused to deliver malicious content.

Implications of this abuse vulnerability in GitHub#

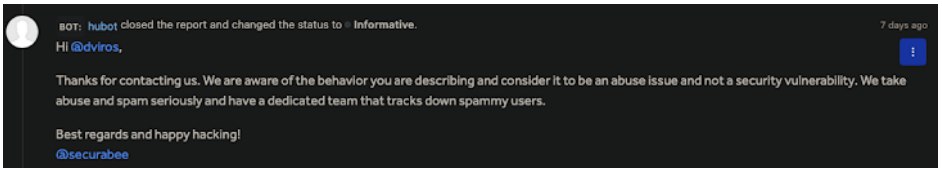
This vulnerability poses a significant risk since it allows threat actors to bypass email security controls and deliver malicious content directly to targeted individuals within organizations. You'll notice that no malicious code was ever deployed. Instead, all of this was accomplished using GitHub's own product in ways it was not intended. The convincing nature of the emails, coupled with the abuse of GitHub's trusted domain, increases the likelihood of successful phishing attacks on busy developers.

Responsible Disclosure and GitHub's Response#

We responsibly disclosed this abuse vulnerability to GitHub through their bug bounty program on HackerOne. Their response:

GitHub's official response (from HackerOne):

> Hi @dviros,
>
> Thanks for contacting us. We are aware of the behavior you are describing and consider it to be an abuse issue and not a security vulnerability. We take abuse and spam seriously and have a dedicated team that tracks down spammy users.
>
> Best regards and happy hacking!
>
> @securabee



GitHub's response indicated that they consider this behavior an abuse issue rather than a security vulnerability, despite these types of abuses account for the vast majority of breaches. They acknowledged the potential for spam and abuse and stated that they have a dedicated team to track down and address such issues.

While GitHub's response is appreciated, we believe that more proactive measures should be taken to mitigate this vulnerability effectively. Some recommendations include:

1. **Enable Privacy Controls:** Allow users, especially enterprise users, to set granular privacy controls and whitelist individuals or organizations that can mention or tag them.
2. **Implement Content Validation:** Leverage language models or other techniques to validate the content of issues, comments, and mentions, particularly when users attempt to mention others for the first time or when the repository has not been previously interacted with by the victim.
3. **Monitor Outgoing Emails:** Implement mechanisms to monitor outgoing emails from notifications@github.com and block any malicious content being delivered.
4. **Streamline Untagging Process:** Provide users with a quick and easy way to "untag" themselves from being mentioned in issues, pull requests, or comments, rather than relying solely on the "unsubscribe" option.

Conclusion#

While GitHub is a powerful and widely adopted platform, this vulnerability highlights the importance of abuse vulnerabilities as the first step taken by threat actors to perpetrate breaches. Simply automatically classifying abuse vulnerabilities as separate from security vulnerabilities discounts their importance in attacks by threat actors.

Dvir Sasson — Director of Security Research at Reco AI
https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEjIsDXLDxfzD9NNyhetG4qGm4HaH5fMaUR37JZKxYOBeVjJl6TZwD9tesXBL_GQk>
BPpSMqwxMYWfsYOWvamq2FoZWVleZYzVcObrq5rQDPP2UJP9wo-XjkXipd_Ad9/s100-rw-e365/reco.png

Found this article interesting? This article is a contributed piece from one of our valued partners. Follow us on Twitter and LinkedIn to read more exclusive content we post.

Source: https://thehackernews.com/expert-insights/2024/05/github-abuse-flaw-shows-why-we-cant.html