Chair of Computer Architecture and Parallel Systems
School of Computation, Information and Technology (CIT)
Technical University of Munich

To What Extent Can We Overcome the Tragedy of the Commons with Digital Goods on the Example of Open-Source Software

Seminar Paper Ethics for Nerds

Matej Svaral✉ **and Yiğit İlk**✉

School of Computation, Information and Technology (CIT), Technical University of Munich

✉ matej.svaral@tum.de
✉ y.ilk@tum.de

September 29, 2023

Abstract — "Tragedy of the Commons" is examined in the context of open-source software (OSS), highlighting the unique challenges it presents. While it does not suffer from issues of traditional commons such as resource depletion, OSS still faces problems like free-riding and dependence risks. The solutions to these challenges include participation in OSS community, responsible use as well as contribution or monetary donations to open-source projects.

1 Introduction

The "Tragedy of the Commons," a concept first introduced by Garrett Hardin in 1968.[1] It highlights the dilemma wherein the use of common goods by individuals, driven by rational self-interest, inevitably leads to depletion and collective detriment. While initially conceived in the context of natural resources such as forests and pastures, this phenomenon has found resonance in various domains, including the digital world. The emergence of widespread open-source software (OSS) projects, such as Linux [2] or Apache [3] providing the backbone of the modern web, present a unique example of a digital common goods. This paper explores the extent to which the Tragedy of the Commons applies in the context of OSS, elaborates on the challenges caused by this tragedy and investigates solutions how to use OSS in an ethical way.

2 Open source software as a digital common good

Open source software (OSS) is software released under one of various licenses granting users the rights to use, modify, and distribute both the software and its source code without restriction. [4] Typically, open source projects consist of the software (the source code and the executable program) and a community of online users contributing to and maintaining it. Both the storage of the code as well as the communication about bugs and feature requests are usually hosted on platforms such as GitHub. [5] Platforms like this are widely favored for their efficient collaboration tools, allowing online users to contribute code snippets (pull requests) approved by maintainers to provide an updated version of the software. [6]

Everybody can access the code and executable program without any restrictions, therefore the software itself is a common good under the non-excludability and non-rivality conditions. While anyone can submit pull requests, the project maintainer retains the authority to approve or reject the contributions, thus the maintenance does not fulfil the previous conditions. This dynamic makes OSS projects imperfect common goods, with only the code and executable components truly comparable to a digital pasture. [7]

3 Application of the tragedy of the commons to OSS

3.1 Original problems

In his original paper, Garrett uses an example of a pasture shared by cattle farmers, facing the question whether a single farmer should add cattle to their herd. Since the benefit of an additional animal is solely received by the farmer while the cost is divided onto all users of the pasture, from the perspective of the farmer, it is economical to add cattle. The problem is that every user of pasture, by the same logic, is rationally incentivized to do the same. As a result, the overall number of cattle inevitably skyrockets, leading to overgrazing of the pasture. This "Tragedy of the commons" is summarized by the following excerpt from his paper:

"Each man is locked into a system that compels him to increase his herd without limit – in a world that is limited. Ruin is the destination toward which all men rush, each pursuing his own best interest in a society that believes in the freedom of the commons" [1]

It is true that individuals may be compelled to "increase their herd" of technology tools with OSS. Though unlike the traditional tragedy where overuse leads to depleted resources, "ruin as the destination" might at most be expressed through technological imperative. If an open-source alternative is better than proprietary software, there is an economical incentive to use it to keep up with the competition. Yet this is not always the case, as the widespread popularity of proprietary software such as Adobe CC or SAP show. The users are not necessarily "locked into this system" when paid software offers better overall value.

While the capacity of the maintainers to approve pull requests is limited, digital goods themselves are infinitely reproducible. Therefore, the notion of a "limited world" does not have a direct effect on the user downloading his copy of the code or executable. There is no such thing as overgrazing when downloading OSS. Actually, the opposite is true, as more users offer more testing opportunities which leads to more stable and secure software. [8]

Due to the different characteristics of OSS as a digital good, we can safely assert that the original tragedy of the commons does not apply.

3.2 Free riding and underfunding of OSS

Although the original tragedy of the commons does not apply to OSS, the problem of free riding still does. Given the right to use the software and its source code without financial obligations, users can access and utilize OSS without contributing to its maintenance costs. This creates a cost dilemma where the burden of fixing bugs falls on the shoulders of maintainers, who cannot pass these costs onto users. Consequently, the incentive for users to free ride is significant, as paying for bug fixes would effectively mean other users receive patches for free. [7] Bigger projects, require more maintenance, but the number of voluntary code contributions and donations does not necessarily keep up with the project size. With high maintenance requirements often surpassing available resources, OSS is often not maintained to current standards. Research shows that 91% of industry software includes open-source components outdated by at least two years, most probably no longer maintained and vulnerable. [9]

Undoubtedly, free riding plays a significant role in underfunding and, in certain instances, inadequate maintenance of OSS. As a consequence, this might increase security risks, as code that receives little or no maintenance remains in widespread use.

3.3 New challenges

With 96% of industry codebases containing open source code [9], the widespread use of OSS has become a double-edged sword. Although having more users and therefore more developers examine the code can help identify bugs, the 2021 example of Log4Shell shows that even a single vulnerability is often very severe as it impacts a massive amount of systems. This vulnerability in the open-source Java library Log4j, resulted in attacks on 40% of all global business networks. [10]

Even today, two years after it has been patched, 11% of systems still run on a vulnerable version of this library. While OSS is easy to install and quickly generate value with, it is the user's responsibility to ensure they are using the latest and most secure version, which introduces the risk of careless use. In contrast to proprietary software, the OSS developers do not have a list of customers, making it challenging to identify its usage and send update notifications, making the use of OSS intransparent. [9] [7]

As with every other software, OSS's value diminishes over time, necessitating constant maintenance. This means that OSS is not a self-sustaining ecosystem like pastures and seas but rather requires ongoing contribution from the community. This very nature of open source, introduces another vulnerability concern if the maintainer approves malicious code.

Collectively these challenges pose serious security risks in an era where our digital infrastructure is heavily reliant on open source technology.

4 Does OSS solve the tragedy of the commons?

Open source software (OSS) overcomes many of the problems associated with original "Tragedy of the Commons". Being a widely used digital common good it allows universal access without depletion. However, it introduces its own set of unique challenges including underfunding due to free-riding and security risks from negligent use. Since OSS is not a self-sustaining system, neglecting the risks can lead to significant consequences. Therefore, OSS requires a set of ethical measures and coordinated efforts to ensure its continued viability and security.

5 Ethical use of OSS

Both free-riding and negligent usage, the two biggest challenges faced by OSS, are fundamentally rooted in user behavior. Addressing these issues primarily involves changing user behavior. While economic incentives do play a role, it is important to note that contributions to OSS from the pure economical self-interest are not directly rational, yet they still occur. Hence, addressing OSS concerns is more about ethics than purely economic incentives.

5.1 Reliance on the community

One of the responses to the original "Tragedy of the Commons" included several empirical examples where the tragedy is successfully mitigated through communication and self-regulation with enforced rules. [11]
Creating a community around each OSS project and efficient communication with the contributors project is vital for its survival. Effective communication within this community helps ensuring that shared maintenance resources are utilized effectively. To avoid negligent use, every user should be familiar about how to use the software and stay informed about the security updates. The the community behind it is usually where they start, hence strong communities and well documented OSS therefore decrease the risk of misuse. With platforms like Github making contribution easier, the least a developer can do is to write a bug report if they discover any. An example of a successful open-source community is Linux. Besides the core software - the Linux kernel - there are various branches called distros with their own communities and enthusiasts contributing.
While the direct impact of joining a community and contributing might not be directly economical, in the long run, a community with a shared goal will make OSS more secure and funded.

5.2 Increase funding

Increasing funding obviously directly addresses this issue of underfunded OSS projects. But how to get the money? Donating and contributing is not only ethically sound but also essential for the projects' sustainability. While there is a first-order consequence of monetary loss and the potential for a free-rider problem, there's also a second-order consequence: When everyone contributes, individuals are incentivized to enhance the software's quality. However, this approach does not always work. Some big companies like Google - that have been partially built on top of OSS - actually make their developers contribute to open-source as part of their workweek as a form of payback. Due to the widespread use of OSS there have even been discussions about state-funded OSS initiatives to ensure the long-term viability, but nothing has been as effective as a simple donation yet. [7]
All of us are most probably willingly or unwillingly using OSS and want to continue to do so - therefore it is only fair to give it some financial support.

5.3 Ensure minimum security standards and correct use

As our reliance on OSS grows, it becomes an essential element of our critical software infrastructure. While the security of the software is the responsibility of the maintainer, installing and using OSS in the proper way and installing the updates remains in the hands of the users. Obviously, the direct solution to this is to shift all the responsibility to the users and not the OSS developers, but this is does not always work. One proposed solution to avoid careless use of OSS is the establishment of a minimal safety standard for OSS to adhere to. A second approach to increase transparency is a "Digital Bill of Materials" listing all dependencies [7], as used for public software contracts in the US.
But the final responsibility to use the OSS as intended and ensure the correct usage lies in the hands of the user.

6 Conclusion

Unlike traditional commons, OSS doesn't suffer from resource depletion, but it does from free-riding and introduces unique software security risks due to negligent use. Being a crucial component of our critical software infrastructure, OSS widespread use and dependence make responsible behavior a matter of ethics. Participation in the OSS community fosters collaboration, transparency, and long-term sustainability, while also decreasing the risk of misuse. There are different approaches to fight the funding problems of OSS project, donations and contributions remain essential

for OSS's sustainability. Establishing minimal safety standards and enhancing transparency can also mitigate careless use, but ultimately, the responsibility for correct usage lies with the users.
Through a strong OSS community, we have managed to a large extend avoid a "Tragedy of the digital commons". But as our digital infrastructure relies heavily on open source technology, addressing the new challenges is imperative to maintain the integrity and security of our digital world.
Thanks to a robust open-source community, we've largely prevented a "Tragedy of the Digital Commons." However, given our heavy reliance on this type of technology, it's crucial to address the new challenges to ensure the integrity and security of our digital world.

References

[1] Garrett Hardin. The tragedy of the commons. *Science*, 162(3859):1243–1248, 1968.

[2] Inc. Linux Kernel Organization. The linux kernel archives. https://www.kernel.org/. Retrieved Sep 29, 2023.

[3] The Apache Software Foundation. Apache http server. https://httpd.apache.org/. Retrieved Sep 29, 2023.

[4] Open Source Initiative. The open source definition. https://opensource.org/osd/. Retrieved Sep 29, 2023.

[5] Inc. GitHub. Github. https://github.com/. Retrieved Sep 29, 2023.

[6] Inc. GitHub. The largest open source community in the world. https://github.com/open-source. Retrieved Sep 29, 2023.

[7] Chinmayi Sharma. Tragedy of the digital commons. *101 North Carolina Law Review 1129 (2023)*, 2022.

[8] Lokshinsardar et al. Quality of open source software: how many eyes are enough? https://blogs.worldbank.org/opendata/quality-open-source-software-how-many-eyes-are-enough. Retrieved Sep 29, 2023.

[9] Inc. Synopsis. [2023] open source security and risk analysis report. https://www.synopsys.com/software-integrity/resources/analyst-reports/open-source-security-risk-analysis.html. Retrieved Sep 29, 2023.

[10] Hannah Murphy. Hackers launch more than 1.2m attacks through log4j flaw. https://www.ft.com/content/d3c244f2-eaba-4c46-9a51-b28fc13d9551. Retrieved Sep 29, 2023.

[11] Elinor Ostrom. The tragedy of the commons. *The New Palgrave Dictionary of Economics*, 2008.