The Prisoner's Dilemma of Open-Source Software Security

Christian Koch, BWI GmbH and Technische Hochschule Nürnberg Georg Simon Ohm

Vulnerabilities in open-source software raise the question of whether governments can trust open source. This article discusses the issue based on the Prisoner's Dilemma.

In March 2024, the software developer Andreas Freund found a potentially malicious backdoor in the Linux utility "xz."[1] The open source library provided functions for data compression and was part of the tool chain of the Linux Secure Shell (SSH). Apparently, the backdoor had been built to compromise the network protocol SSH and gain access to machines running it. Empirical evidence suggests that the breach could be the result of a targeted government operation. One or more developers seem to have infiltrated the xz project and created the backdoor. Hacker groups performing such operations are called advanced persistent threats (APTs). One infamous example is the group APT28, also known as "Fancy Bear." Allegedly, the Russian hacker collective is responsible for various assaults on journalists, companies, and government agencies.[2]

As the "xz backdoor" was not the first attack targeting open-source software (OSS), it has fueled an ongoing debate about whether governments can trust open source. One influential essay on this issue was published in 1999 by Bruce Schneier.[3] According to his article, "[p]ublic security is always more secure than proprietary security." To this day, a lively debate has been raging whether this is true. Attacks such as the "xz backdoor" cast doubt on the hypothesis. This essay analyzes the question based on a model familiar to economists and military strategists: the prisoner's dilemma.

COOPERATION VERSUS DEFECTION

Originally, the prisoner's dilemma was formulated by Meril Flood and Marvin Dresher and later formalized by Albert

Digital Object Identifier 10.1109/MC.2024.3415868
Date of current version: 23 September 2024

EDITOR DIRK RIEHLE
Friedrich Alexander-University of Erlangen Nürnberg;
dirk.riehle@fau.de

Tucker[4] (p. 154). Today, it is one of the foundations of an academic field called *game theory*. This branch of mathematics analyzes problems in which two actors play a "game" and have the choice to either cooperate or "defect" against each other. Depending on their choice, the players receive a certain payoff after a game.

The prisoner's dilemma is simple to explain: Let us imagine that there are two gangsters (A and B) who have committed a crime and are caught by the police. Yet the police have no substantial evidence against the suspects. Both gangsters are interrogated separately and have no knowledge of the other's testimony. If both suspects remain silent, they will be imprisoned only for one year, due to the lack of evidence. However, if one of them testifies while the other remains silent, the traitor is released, while the betrayed is sentenced to 10 years in prison. Finally, if both gangsters testify, they are punished with five years each. Table 1 illustrates the payoffs of the game. (Since the penalties are adverse for the gangsters, the payoffs are negative.)

Let us consider some examples based on Table 1. If player A stays silent while player B testifies, A is charged with 10 years, while B is released (–10/0). In case that both A and B testify, they are imprisoned for five years (–5/–5). Each one is sentenced one year, if A and B both stay silent (–1/–1). In game theory, the prisoner's dilemma serves as a metaphor for scenarios in which two players can achieve a global optimum only if both cooperate. Precondition for this is mutual trust. Whenever a player cannot trust the other party, it is rational to defect. As otherwise, he/she faces the worst possible payoff.

NUCLEAR WARFARE AND OSS

Historically, the prisoner's dilemma was used to analyze strategies of nuclear warfare[4] (p. 155). Let us imagine two countries (A and B). Together they achieve a global optimum, when neither side possesses atomic bombs. In this case, both countries can live without the risk of being eliminated by the other's nuclear weapons. However, following the prisoner's dilemma, the worst outcome for one country is achieved, if its opponent owns atomic bombs exclusively. In this case, the nuclear power might be tempted to use it in the event of a conflict. This is what happened in World War II. Therefore, if two countries cannot trust each other, it is rational for both to possess atomic bombs. Strategies of nuclear deterrence build on this rationale. (Whether they are justifiable from a moral perspective is another discussion.) In game theory, the approach is referred to as a "dominant" strategy. It leads to the best possible outcome for one player, regardless of the opponent's choice.

Core hypothesis of this article is that governments face a comparable situation with OSS security. Table 2 shows a variant of the prisoner's dilemma representing this problem.[a] To reflect the different scenario, the game is based on positive and negative payoffs. However, the exact numbers are not crucial, it is the principle that matters. Let us imagine two states (A and B). Both have the choice of either compromising open-source software or staying passive. Together, they achieve a global optimum when neither side tries to compromise OSS (+1/+1). In this case, each country can use open-source software securely and does not have to fear an attack by its counterpart. Yet this requires trust between the players.

[a]To be precise, the formally most suitable model would be an iterated n-prisoner's dilemma. This article uses the basic variant for the sake of simplicity. Focus of the essay lies on the principle, not the details.

FROM THE EDITOR

Welcome back to our column on open source! In this month's instance, Christian Koch of BWI GmbH, a public IT services provider owned by the German Ministry of Defence takes a look at security and open-source software, a topic we have visited in the past and will revisit in the future. In other words: This topic is burning hot. The novelty of Christian's article is to interpret open-source security from a government's perspective using the well-known prisoner's dilemma theory. For one, not just happy hacking, but happy strategizing as well!—*Dirk Riehle*

TABLE 1. Payoff matrix of the Prisoner's Dilemma.

A/B	B stays silent	B testifies
A stays silent	(–1/–1)	(–10/0)
A testifies	(0/–10)	(–5/–5)

TABLE 2. Payoff matrix of the OSS dilemma.

A/B	B remains passive	B compromises OSS
A remains passive	(+1/+1)	(–1/+1)
A compromises OSS	(+1/–1)	(0/0)

As with the prisoner's dilemma, the worst situation for a country in the game is achieved if it remains passive while the opponent compromises OSS. In the event of a conflict, the defecting state can use the backdoors for an attack on the other side. In absence of mutual trust, it is therefore rational for both states in the game to compromise open source projects. This way, they achieve a balance of power reflected in equal payoffs (0/0). When only one country compromises OSS, it gains superiority over its counterpart. In Table 2, this situation is expressed by opposing positive and negative payoffs.

There is a difference between the game outlined in Table 2 and a so-called *free-rider problem*. Free-rider problems represent scenarios where one player benefits from the work of others without contributing. We can model such a game also as a variant of the prisoner's dilemma (or n-prisoner's dilemma) with strictly positive payoffs.[5] If a free rider defects, the betrayed players gain less, but still win. Merely they are denied the opportunity of achieving higher payoffs. In comparison, if one country compromises OSS with the aim of an attack, the targeted state may find itself in a worse position than before. In other words, players risk a win–lose situation in the game specified in Table 2. In absence of trust, defection is a rational choice. An alternative to defection is to try to change the game by reducing its negative payoffs (represented by –1). Attacking countries then could still create a relative advantage, albeit with limited destructive impact. How to realize such a protective game change is discussed next.

ZERO-TRUST SECURITY

Following the logic of the prisoner's dilemma, the answer to the question of whether governments should trust open-source software is no. At least not if they cannot trust each other. Vulnerabilities such as the "xz backdoor" and APTs like Fancy Bear speak a clear language. But if governments cannot trust open-source software, should they replace it with closed-source alternatives? Following the idea of this essay, the answer is no again. Just as we can use the prisoner's dilemma to analyze OSS security, we can do the same for closed-source software (CSS). While CSS may provide higher hurdles for infiltrators, OSS offers more transparency. Vulnerabilities like the "xz backdoor" have been found precisely because their source code was open.

In a contested environment with APTs operating, governments should neither trust OSS nor CSS. Clearly, there are breaches in open-source software. But the same is true for closed-source products. One example for the latter is the Stuxnet computer worm, used to attack Iranian nuclear facilities via the Siemens S7 software running on Windows.[6] By destroying industrial devices, the malware showed that damage caused by cyberattacks is not limited to information systems. One related example is the EternalBlue vulnerability, allegedly exploited by the U.S. National Security Agency (NSA). According to Wicker, the agency "had lost control of the assets it had developed to take advantage of the vulnerability" and the exploit was used by an APT to implement the WannaCry ransomware.[7] In such a hostile playing field, zero trust is the only sustainable strategy.

In the context of information security, zero-trust models assume that attackers are present in every environment and that internal environments are no different—or more trustworthy—than external environments.[8] When implementing a zero-trust approach, government agencies must not grant implicit trust and continuously analyze risks to their assets. In response, protective measures are taken to mitigate the identified risks. Examples of protective measures are minimizing access to resources and the continuous authentication and authorization of users. Following the philosophy of zero trust, the question of whether open source is more or less secure than closed source misses the point. For APTs, it is equally rational to compromise OSS and CSS. Governments should trust neither one.

In an insecure world, zero-trust security models are a rational choice for both government agencies and businesses. Establishing an effective cyber defense helps to mitigate the vicious circle of the prisoner's dilemma. By improving defenses, government entities can limit the potential damage caused by attackers and reduce the game's negative payoffs. When it comes to risk, diversification is a powerful measure to improve protection and to change the game. Instead of putting all eggs in one basket, it is preferable for government agencies to combine different software products from multiple vendors. Whether these are open or closed source is not the key question from a security perspective.

When implementing a zero-trust approach, government entities should combine redundant and independent security measures. In military strategy, this principle is called *defense in depth*. When the first line of defense is breached, a second line takes effect, then a third, and so on. Transferring this pattern to information systems builds on the hope that not every software is compromised—at least not by the same APT.[b] Compared to other dimensions of warfare, models such as zero trust offer an advantage in the cyber domain: They allow for strategies of a nonoffensive defense. That means that governments can protect their information architecture without threatening or actively harming other states. In areas like nuclear warfare, this goal is much harder to accomplish.

However, limiting the potential damage from adversaries does not necessarily stop them from gaining a relative superiority in the game. To achieve that, it is sufficient for them to realize

[b]Imagine a system protected by five security layers, and each of them is breached with a probability of 20%. Assuming independence, the likelihood of all five layers being compromised shrinks to 0.03% ($0.2^5 * 100$).

higher profits than their opponents. Whether governments should respond to this with offensive cyber operations is a question of morality and not the subject of this essay. The key lesson from this article is that zero trust must be the guiding principle of every cybersecurity strategy. In a contested environment where the prisoner's dilemma prevails, it is an imperative toward resilience. **C**

REFERENCES

1. J. Corbet, "A backdoor in xz." LWN. Accessed: May 11, 2024. [Online]. Available: https://lwn.net/Articles/967180/
2. B. Jensen, V. Brandon, and M. Ryan, "Fancy bears and digital trolls: Cyber strategy with a Russian twist," in *Military Strategy in the 21st Century*, Evanston, IL, USA: Routledge, 2020, pp. 58–80.
3. B. Schneier, "Schneier on security." Schneier. Accessed: May 11, 2024. [Online]. Available: https://www.schneier.com/crypto-gram/archives/1999/0915.html#OpenSourceandSecurity
4. L. Freedman, *Strategy: A History*. London, U.K.: Oxford Univ. Press, 2015.
5. R. Hardin and C. Garrett, *The Free Rider Problem*. Stanford, CA, USA: Stanford Univ. Press, Oct. 2020.
6. T. M. Chen and A.-N. Saeed, "Lessons from Stuxnet," *Computer*, vol. 44, no. 4, pp. 91–93, 2011, doi: 10.1109/MC.2011.115.
7. S. B. Wicker, "The ethics of zero-day exploits-the NSA meets the trolley car," *Commun. ACM*, vol. 64, no. 1, pp. 97–103, 2020, doi: 10.1145/3393670.
8. S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture, special publication (NIST SP)," National Institute of Standards and Technology, Gaithersburg, MD, USA, 2020. Accessed: Jun. 24, 2024. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-207, https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=930420

CHRISTIAN KOCH is an enterprise lead architect at BWI GmbH and a lecturer at the Nuremberg Institute of Technology Georg Simon Ohm. Contact him at christian.koch@bwi.de.

Digital Object Identifier 10.1109/MC.2024.3449494