An Analysis of the Effects of Open-Source Cyber Weapons

A Research Paper submitted to the Department of Engineering and Society

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Chase Hildebrand

Spring 2024

Advisor

Kent Wayland, Department of Engineering and Society

Introduction

In a world dominated by digital connectivity, cybersecurity is rapidly emerging as a serious concern that is ubiquitous across all industries. As our reliance on information technology systems grows, so do the threats that seek to exploit them. In 2023, cybercrime cost the United States a total of $12.5 billion based on reported losses (Federal Bureau of Investigation, 2023). Just as defenders rely on tools to protect themselves, attackers rely on tools to attack others; among the most crucial of these tools are command and control (C2) frameworks. In recent years, there has been an uptick in the number of these frameworks that are open source and released by security experts themselves (*C2Matrix*, n.d.), which presents an interesting question: should experts be releasing cyber weapons for their adversaries to freely use against them?

This paper seeks to investigate the complex dance between how different actors use open-source C2 frameworks to understand the extent to which building and releasing these malicious software is harmful or helpful to the security of our society. Gaining insight into this question is crucial, as it ensures that the work of security researchers in building these tools maximizes its positive impact rather than inadvertently causing harm. If these tools prove beneficial, we as a community should encourage researchers to release them; if they are harmful, then we should discourage their dissemination.

Background

When an attacker wants to attack a system, they typically follow a series of steps called the "cyber kill chain". Traditionally, the cyber kill chain contains seven steps: reconnaissance, weaponization, delivery, exploitation, installation, command and control, actions on objectives

(Skopik & Pahi, 2020). First, an attacker starts by doing reconnaissance on their target. Their goal is to find out as much as possible about their target's network, but also social factors like personal information about employees, physical information like office locations and lock types, and any other information that could help them identify a vulnerability in their target. A vulnerability is a flaw in a system that can be exploited to gain access. Once they have identified a vulnerability, they develop an exploit to take advantage of the vulnerability and send it to their target. An exploit is a tool or technique that takes advantage of a vulnerability to gain access to or otherwise compromise a system. The act of developing an exploit, delivering it to a target, and activating it is called weaponization, delivery, and exploitation respectively. After gaining access to a system, the attacker will install malware and set up a communication channel between the attacker and the malware running on the target's systems, which will let the attacker take actions on the target. This communication between the attacker and the victim is called "command and control" or C2 for short. After setting up command and control, the attacker will use the malware to carry out the objectives they have. These could include exfiltrating personally identifiable information or company secrets, deploying ransomware, or denying service.

Unlike most of the other steps in the cyber kill chain, command and control is continuous and must stay hidden to evade detection, even when defenders are looking. To do all of this, attackers usually utilize a C2 framework, which is a collection of tools that allow attackers to manage their access. These frameworks usually include three components: the beacon (also known as an implant), the C2 server, and the client or console. The beacon is the malware running on a target's system—it will occasionally call back to the C2 server to fetch tasks to perform and return the results. An attacker uses the C2 client to send tasks to the server for distribution to the beacons. In addition to these basic features, many C2 frameworks support a wide range of more advanced features such as anti-detection, automatic malware generation, and support for "multiplayer mode" where multiple attackers can submit tasks to the beacons concurrently.

Within the past ten years, people in cybersecurity have started developing their own command and control frameworks and started either making them commercially available or publishing the source code on sites like GitHub. Because these frameworks are open source, anyone can download the source code and use them free of charge. Some examples of these projects include Sliver (BishopFox, 2019/2024), Merlin (Tuyl, 2017/2024), Realm (Spellshift, 2022/2024), Havoc (*HavocFramework/Havoc*, 2022/2024), Mythic (Thomas, 2018/2024), and PowerShell Empire (BC Security, 2019/2024). Another notable project, Cobalt Strike, while paid and not open source, is widely used and commercially available to anyone for purchase. Recently, these frameworks have become quite sophisticated, including features from pivoting (using one infected host to attack another) to being able to manage hundreds of beacons at once.

Methods

To conduct this analysis, I first collected information about the different actors involved and how they interact with open-source C2 frameworks. To gather the information about these relationships, I reviewed incident response reports, cybersecurity news articles, blog posts made by security professionals, and scientific research papers.

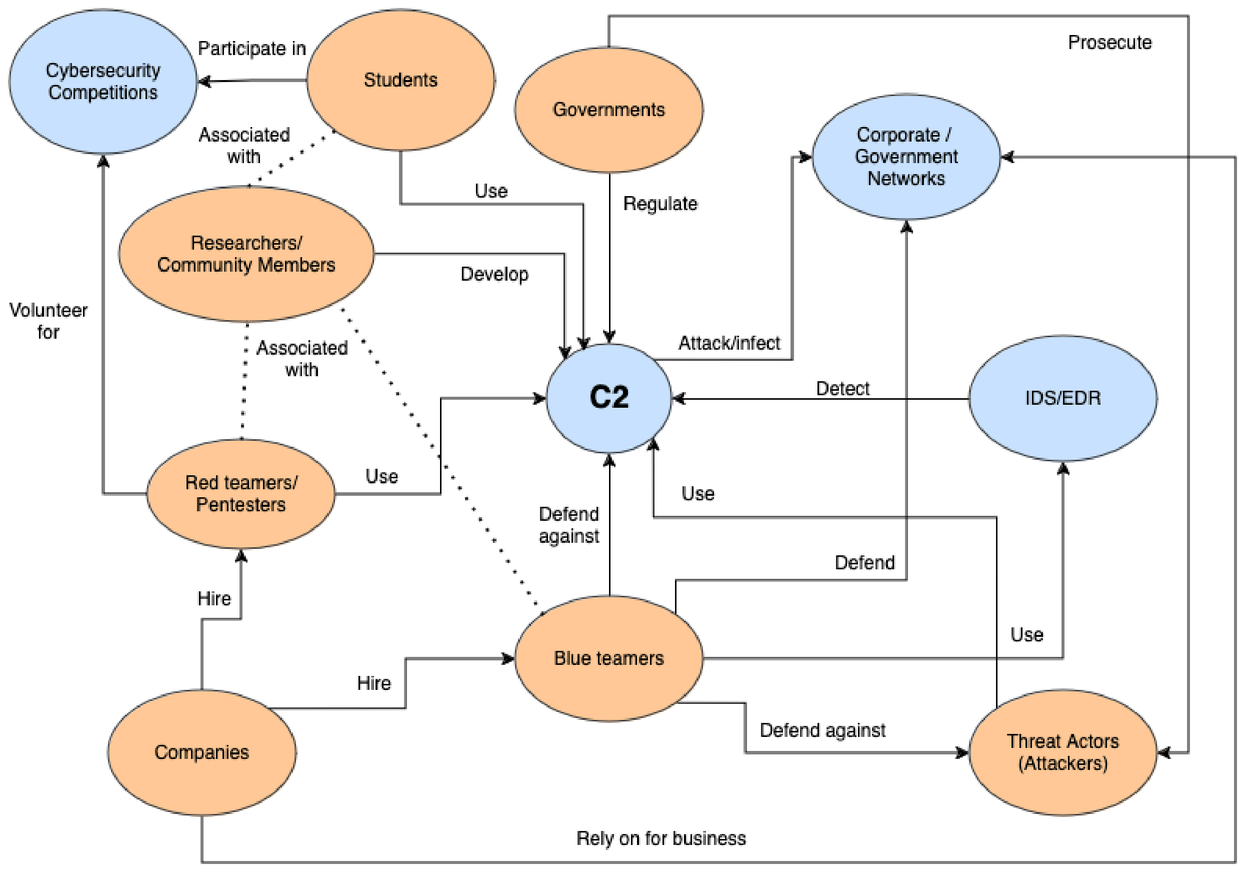
To analyze the complex relationships between these groups, I used actor-network theory (ANT). ANT examines how networks of both human and non-human actors shape a system in society. It views human and non-human entities as having an equally important role in the interconnected network they are a part of.

Results

There are numerous reports of threat actors using open-source tools to attack networks. In November of 2023, Cloudflare, a major internet delivery company was attacked. The threat actor used leaked credentials to log into a public facing Jira instance and installed Sliver C2 to establish persistence on the network (Prince et al., 2024). Even sophisticated, well-resourced attack groups—which are typically backed by nation-states and known as advanced persistent threats (APTs)—such as APT-33 and APT-19, have been caught using PowerSploit, Powershell Empire, and PoshC2 to attack their targets (Nelson & Kettani, 2020). Microsoft reports that they have observed nation-state threat actors and ransomware groups using Sliver (Microsoft, 2022). In January 2024, a likely Russian APT sent a phishing email containing a zip file that had a LNK file which executed an open-source project called HTTP-Shell (Cluster25, 2024; GM, 2023/2024). Additionally, ransomware groups have made heavy use of open-source C2 frameworks too. For example, Mandient reports that "beginning in late 2019, FIN12 appeared to reduce their reliance on TRICKBOT and began to use publicly available tools for the post-compromise stages of their intrusions" (Mandiant, 2023, p. 12).

Threat actors are not the only ones using these open-source C2 frameworks. In the past 20 years, cybersecurity competitions like the Information Security Talent Search (ISTS), the Collegiate Penetration Testing Competition (CPTC), and the Collegiate Cyber Defense Competition (CCDC), have been gaining popularity among college students. These competitions are designed for students to learn cybersecurity in a realistic and hands-on environment (Conklin, 2005). At the CCDC National Championship, the red team uses Sliver, Merlin, and Realm C2 frameworks to emulate threat actor behavior (Borges, 2021). In addition to competitions, students have also used this software for certain cybersecurity coursework to learn how to think like an attacker (OConnor, 2022).

Analysis



Enumeration of actors

There are several actors involved in the actor network induced by open-source C2 frameworks.

Open-source command and control frameworks (inanimate)

Open-source command and control frameworks are the inanimate actor at the center of the network. They are used by threat actors, cybersecurity professionals, and students and interact with inanimate actors such as EDRs and IDS's and corporate networks.

Government and corporate networks (inanimate)

There are the networks used by companies or governments that are the target of cyber-attacks. These networks often contain sensitive information that attackers are after. They are defended by security professionals and tested for vulnerabilities by penetration testers and red teams.

Blue teams (human)

Blue teams are teams of professionals whose job is to monitor and defend corporate and government networks. To do their jobs more effectively, blue teams are typically equipped with software that is designed to flag suspicious activity or automatically remove malware. Some examples of this software include Security Event and Incident Management (SIEM) systems, Endpoint Detection and Response (EDR) systems, and Intrusion Detection Systems (IDS). SIEMs aggregate logs and security events in real time in one place to give defenders visibility across a network. EDRs detect malicious activity on a host and automatically respond to it by neutralizing the threat. IDSs search for suspicious activity on a system and send it to a SIEM for analysis. Using these tools and their own skills and experience, blue teams constantly monitor the network to search for threats. If a blue team identifies suspicious activity, they are responsible for triaging the situation, threat hunting, and neutralizing the threat on the network.

Being able to accurately assess how well these tools are working is critically important to defenders.

Red teams and penetration testers (human)

Red teams and penetration testers are teams of professionals that are hired by companies to conduct security assessments. They try to break into the company's network to simulate a real attack and help the company find weaknesses in their network before they can be exploited by threat actors. Like how blue teams use defensive tools, red teams also use extensive tooling to make their job easier. One of the most important tools in their toolbox is command and control frameworks.

Rather than spending time developing their own command and control framework from scratch, many red teamers utilize publicly or commercially available ones to make their jobs easier. Several of these C2 frameworks, like Sliver and Merlin, also have built in functionality to record all the actions a red teamer takes during an engagement, so they can easily generate a report for the customer showing them exactly where their defenses could be improved.

Students (human)

Students, in this context, will refer to the group of high school and university students who are learning about cybersecurity and primarily focusing on improving their skills and knowledge. As people who are just starting out in cybersecurity, it is helpful to have a collection of tools that are freely available and have a friendly learning curve.

Cybersecurity competitions (inanimate)

Cybersecurity competitions provide a way for students to learn cybersecurity in a hands-on fashion without any risk. There are many competitions available for students to participate in that provide an engaging, hands-on, and realistic environment for them to learn about cybersecurity, while simultaneously building talent in the field. Some of these competitions include Cyber Patriot, Collegiate Penetration Testing Competition (CPTC), and the Collegiate Cyber Defense Competition (CCDC). These competitions serve as an inanimate actor in the system, but further unpacking them is outside the scope of this research project.

In defense competitions, like CCDC, the competition organizers employ teams of volunteer red teamers to simulate attacks on the student networks. These red teams heavily rely on open-source C2 frameworks to accurately simulate attacker behavior. In offense competitions, which require students to attack a mock network, student teams find open-source C2 frameworks helpful when mounting their attacks.

Effects of the relationships between the actors

Because people are sharing implementation of new techniques made or discovered with each other, new advancements propagate through the actor network extremely quickly. For example, presume a security researcher publishes a paper on a new way to bypass Microsoft Defender Antivirus. A red team may implement this as a feature in an open-source tool and publish it for other red teams to use. Because more penetration testers are using this technique, blue teams will understand how to detect and respond to this threat. Meanwhile, as use of this new technique increases, IDS/EDR systems will get better at recognizing and flagging it. This means that blue teams are more likely to catch closed-source, threat-actor-made tools that also use this technique.

Unfortunately, this also means that threat actors will have access to the tool and use it against their targets. However, because the blue teams and IDS/EDR products know how to catch this technique, the open-source tools employed by threat actors will have reduced effect. This being said, advanced threat actors may modify the code or develop additional software to sneak the malware by defenses without blue teams knowing. One such example is the KrustyLoader, whose purpose is to deliver a Sliver beacon without being detected (Letailleur, 2024).

This mutual shaping between the sophistication of open-source C2 frameworks and the quality and expectations of defenses is a careful balance that leads to increased quality of our defenses, but also lower skill required to perform a sophisticated attack.

A world without open-source C2 frameworks

Examining the network diagram, we can imagine a world where C2 frameworks were not present. This could be due to a variety of reasons from a general discouragement within the community to government regulations preventing people from distributing cyber weapons to the public. To model this scenario, we will remove the open-source C2 node from the network and redraw the relations based on how each actor would react.

Red teams and penetration testers would not be able to rely on open-source tools to do their jobs and would have to write their own. This would take up a significant amount of time and would either significantly increase costs or discourage security testing from happening at all. Looking at the interaction between students and open-source C2 frameworks, students would not benefit from the realistic adversary simulation that they provide. As a result of the lack of realistic engagements that C2 frameworks help provide, the workforce would be less skilled overall in defending against cyber-attacks. This leads to less experienced red and blue teams. The good blue teams would be the ones who have dealt with incidents in real life or test their defensive systems and abilities on either their own adversary emulation tooling or live malware collected from the wild, which has its own set of issues.

As a result of these issues, many actors are incentivized to build C2 frameworks independently anyway, even if none are publicly available. Because everyone is working on these independently, it is significantly less efficient. Additionally, the community doesn't get the benefit of being able to share ideas between groups. Based on the actor network, preventing or limiting the distribution of C2 frameworks publicly has the result of segmenting the security community and incentivizing actors to build them themselves anyway, resulting in replicated and less innovative work. In this world, the C2 inanimate actor would be significantly more fragmented which would increase the time it takes for developments of new techniques to propagate through the network.

Another potential response to this situation rather than simply removing C2 from the equation could be to require a background check or similar screening before acquiring cyber weapons. Said screening could require researchers to demonstrate that they have a background as a security professional or researcher and are well established in the community or are a student learning cybersecurity. This would introduce additional complexity for each human actor that works with the C2 frameworks; the red teams, penetration testers, security researchers, and defensive software developers would need to directly interact with the government too. This introduces significant complexity and inefficiency. For example, take the perspective of a student who wants to use Sliver to learn how to hunt for malware running in memory. Learning cybersecurity is already intimidating as is and having to go through an approval process would add more intimidation and may discourage students to pursue a career in cyber. Additionally, even with a protective process like this in place, there is still a risk for students to exploit C2 frameworks anyway. There have been numerous reports of teenagers breaking into large companies, like the 18 year old Oxford student who hacked Uber and Rockstar Games in 2022 (Tidy, 2023).

A process like this could make sense for C2 frameworks that implement publicly unknown or unreleased techniques to bypass defense products. A red teamer's job is to accurately assess the security of their client, and sometimes that means simulating an advanced attacker or APT who has capabilities to develop exploits and tools that are unknown to the public. They might not have the time or programming experience to develop these exploits themselves so it would make sense to buy them from a company that specializes in developing this type of software. In this case, it may make sense to restrict this sale to those who are trusted professionals so they do not fall into the wrong hands, as the techniques they use to circumvent defense tools are intentionally kept private and likely unpatched by the makers of said defense tools. This seems reasonable, but looking back at the relationship between threat actors and regulatory bodies sheds light on an important detail: threat actors may start attacking this system. Threat actors also do not abide by licenses, regulations, or laws. If just one threat actor gets their hands on this software, they could share it with all of their associates or sell it on the black market completely unrestricted for a drastically reduced price. Because the public availability of malicious software shapes the skill of defenders and sophistication of defense tools, its absence in defenders' arsenal will create an imbalance between them and the attackers who possess the malicious software.

In addition to screenings being inconvenient for a vast majority of users, it would likely not be effective: if the background requirements are not strong enough, then the threat actors could also get their hands on the software, but if they are too strict then they will be inaccessible to certain benign groups, especially those who are not as established in the field. Restricting powerful tools to a group would seem to work in the short term, but as soon as threat actors get a hold of these tools they are in an advantageous position against defenders. By forcing us to consider relationships between parties that we may not have otherwise considered, ANT helps reveal potential flaws in such a system.

Conclusion

While the distribution of open-source C2 frameworks has given attackers a boost in attacking their targets, they have also given defenders tools to teach others, share knowledge, and test their defenses. Banning or discouraging the distribution of these open-source projects would deprive threat actors access to a pool of dangerous hacking tools, but it would also have the more subtle effect of slowing down development of robust defensive tools, making penetration tests more inefficient, and inhibiting cybersecurity education for newcomers to the field.

This paper focuses purely on open-source C2 frameworks, but analyzing the mutual shaping between the broader field of open-source malware and exploits and society remains an area for future work. These could include tools like Mimikatz (DELPY, 2014/2024), SSH-Snake (Rogers, 2023/2024), and NetExec (byt3bl33d3r et al., 2015/2024) which are designed to attack systems but are not purely command and control frameworks. Another area for future work would be to analyze the mutual shaping between threat actors and open-source software in general. This is especially relevant after the XZ attack, where an unknown threat actor put a backdoor in a piece of widely used open-source software (Freund, 2024). Finally, this paper treats cybersecurity competitions as an actor in the actor network, but further unpacking cybersecurity competitions into their components is an area for further research.

With the reports of cybercriminals abusing open-source C2 frameworks, slowing down their distribution seems like an obvious choice. However, using ANT to look into the sociotechnical system constructed by their public availability, it is clear that this issue is significantly more complex and requires careful thought. There is one advantage that we defenders have against our adversaries: our rapid propagation of information between each other. Though initially non intuitive, with analysis with ANT it becomes more apparent that keeping open-source command and control frameworks around actually makes the world a safer place.

References

BC Security. (2024). *BC-SECURITY/Empire* [PowerShell]. BC Security. https://github.com/BC-SECURITY/Empire (Original work published 2019)

BishopFox. (2024). *BishopFox/sliver* [Go]. Bishop Fox. https://github.com/BishopFox/sliver (Original work published 2019)

Borges, D. (2021, May 6). NCCDC 2021 Red Team Review. *LockBoxx*. http://lockboxx.blogspot.com/2021/05/nccdc-2021-red-team-review.html

byt3bl33d3r, mpgn_x64, Hallenbeck, M., & zblurx. (2024). *Pennyw0rth/NetExec* [Python]. Pennyw0rth. https://github.com/Pennyw0rth/NetExec (Original work published 2015)

C2Matrix. (n.d.). Google Docs. Retrieved May 6, 2024, from https://docs.google.com/spreadsheets/u/1/d/1b4mUxa6cDQuTV2BPC6aA-GR4zGZi0oo PYtBe4IgPsSc

Cluster25. (2024, January 30). *The Bear and The Shell: New Campaign Against Russian Opposition -*. https://www.duskrise.com/2024/01/30/the-bear-and-the-shell-new-campaign-against-russi an-opposition/

Conklin, A. (2005). The use of a collegiate cyber defense competition in information security education. *Proceedings of the 2nd Annual Conference on Information Security Curriculum Development*, 16–18. https://doi.org/10.1145/1107622.1107627

DELPY, B. (2024). *Gentilkiwi/mimikatz* [C]. https://github.com/gentilkiwi/mimikatz (Original work published 2014)

Federal Bureau of Investigation. (2023). *Internet Crime Report*. https://www.ic3.gov/Media/PDF/AnnualReport/2023_IC3Report.pdf

Freund, A. (2024, March 29). *oss-security—Backdoor in upstream xz/liblzma leading to ssh server compromise*. https://www.openwall.com/lists/oss-security/2024/03/29/4

GM, J. (2024). *JoelGMSec/HTTP-Shell* [PowerShell]. https://github.com/JoelGMSec/HTTP-Shell (Original work published 2023)

HavocFramework/Havoc. (2024). [Go]. Havoc Framework. https://github.com/HavocFramework/Havoc (Original work published 2022)

Letailleur, T. (2024, January 29). *KrustyLoader—Rust malware linked to Ivanti ConnectSecure compromises*. Synacktiv. https://www.synacktiv.com/en/publications/krustyloader-rust-malware-linked-to-ivanti-co nnectsecure-compromises

Mandiant. (2023). *FIN12 group profile: FIN12 prioritizes speed to deploy ransomware against high-value targets*. https://www.mandiant.com/resources/reports/fin12-group-profile-fin12-prioritizes-speed- deploy-ransomware-against-high-value-targets

Microsoft, M. (2022, August 24). *Looking for the 'Sliver' lining: Hunting for emerging command-and-control frameworks*. Microsoft Security Blog. https://www.microsoft.com/en-us/security/blog/2022/08/24/looking-for-the-sliver-lining- hunting-for-emerging-command-and-control-frameworks/

Nelson, T., & Kettani, H. (2020). Open Source PowerShell-Written Post Exploitation Frameworks Used by Cyber Espionage Groups. *2020 3rd International Conference on Information and Computer Technologies (ICICT)*, 451–456. https://doi.org/10.1109/ICICT50521.2020.00078

OConnor, T. (2022). HELO DarkSide: Breaking Free From Katas and Embracing the Adversarial

Mindset in Cybersecurity Education. *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1*, *1*, 710–716. https://doi.org/10.1145/3478431.3499404

Prince, M., Graham-Cumming, J., & Bourzikas, G. (2024, February 1). *Thanksgiving 2023 security incident*. The Cloudflare Blog. https://blog.cloudflare.com/thanksgiving-2023-security-incident

Rogers, J. (2024). *MegaManSec/SSH-Snake* [Shell]. https://github.com/MegaManSec/SSH-Snake (Original work published 2023)

Skopik, F., & Pahi, T. (2020). Under false flag: Using technical artifacts for cyber attack attribution. *Cybersecurity*, *3*(1), 8. https://doi.org/10.1186/s42400-020-00048-4

Spellshift. (2024). *Spellshift/realm* [Rust]. Spellshift. https://github.com/spellshift/realm (Original work published 2022)

Thomas, C. (2024). *Its-a-feature/Mythic* [JavaScript]. https://github.com/its-a-feature/Mythic (Original work published 2018)

Tidy, J. (2023, December 21). *Lapsus$: GTA 6 hacker handed indefinite hospital order*. https://www.bbc.com/news/technology-67663128

Tuyl, R. V. (2024). *Ne0nd0g/merlin* [Go]. https://github.com/Ne0nd0g/merlin (Original work published 2017)