DARKOWL

VERSION CONTROL SYSTEMS (VCS) AND SOFTWARE SUPPLY CHAIN RISK

Research from DarkOwl analysts continues to indicate that software development and engineering tools are a viable exploitation vector

April, 2022

TABLE OF CONTENTS

Intro

In recent months, DarkOwl has observed a significant increase in instances of malware developers mentioning or discussing direct attacks to the international software supply chain. In many cases, this chatter was centered around plans that involved targeting popular open-source software developer repositories like Github and Bitbucket, as well as associated software digital support infrastructure.

Exploitation of software-build processes and code repositories facilitates wider, more-catastrophic distribution of malware and enterprise-level software compromise. By poisoning software development, update processes, and link dependencies, a threat actor's malicious code can be potentially distributed to thousands of users without need for social engineering, e-mail compromise, or drive-by-download malware delivery mechanisms.

Developers and security researchers around the world have been equally appalled and conflicted by the recent intentional sabotage of an open-source software package. Many are particularly concerned about the reputational damage these incidences cause to the open-source software development movement.

Despite general widespread sentiments against Putin's invasion of Ukraine, the open-source software development community has marked RIAEvanglist's NPM package as malicious, because this individual chose to deploy malware in the digital supply chain ecosystem.

"THIS CODE SERVES AS A NON-DESTRUCTIVE EXAMPLE OF WHY CONTROLLING YOUR NODE MODULES IS IMPORTANT. IT ALSO SERVES AS A NON-VIOLENT PROTEST AGAINST RUSSIA'S AGGRESSION THAT THREATENS THE WORLD RIGHT NOW. THIS MODULE WILL ADD A MESSAGE OF PEACE ON YOUR USERS' DESKTOPS, AND IT WILL ONLY DO IT IF IT DOES NOT ALREADY EXIST JUST TO BE POLITE."

— 'peacenotwar' source code description

In one recent example that occurred mid-March 2022, a person responsible for maintaining an NPM package – a widely used package manager for the JavaScript programming language – showcased how potentially powerful supply chain attacks on software development and components can be. This individual, an open-source software developer known as RIAEvangelist, intentionally embedded malware in the latest stable release of a popular repository called node-ipc out of protest for Putin's atrocities against Ukraine. The malware is officially labeled 'peacenotwar' and deploys with a readme file titled **WITH-LOVE-FROM-AMERICA.txt,** and notably only is triggered to install on devices with a Belarus or Russia geo-located IP addresses.[1]

Exploiting Version Control Systems (VCS) and poisoning supply chains is not a new threat vector. In 2021, the Kaseya ransomware attack – via a simple malicious software update pushed to thousands of users by notorious ransomware gang, REvil – highlighted the extensive threat to software supply chains and cloud-based commercial software repositories.[2]

December 2020's Solarwinds's attack similarly inspired international concern for the integrity of commercial enterprise software and underscored the need for widespread implementation of zero trust architectures.[3]

Another example of a threat actor group exploiting digital supply chain vulnerabilities is the hacking group LAPSUS$. The increasingly active group most recently announced that they had acquired privileged access to digital authenticator Okta's networks via a support engineer's thin client. The result of Okta's compromise exposed significant intelligence findings and highlights the overarching risks at stake to any software development and operational lifecycle.

Below is a brief summary of how LAPSUS$ leveraged supply chain exploits to compromise global software company Okta – which will be discussed in more detail later in the report:

- **LAPSUS$ most likely gained access to Okta using credentials purchased on the deep web marketplace: Genesis Market[4], proving the underground continues to feed criminal empires.**
- **Amazon Web Services (AWS) credentials and code repository tokens were likely shared in cleartext in company Slack messaging systems that LAPSUS$ then utilized to move laterally through peripherally associated digital infrastructure.**
- **LAPSUS$ clearly stated they were not interested in Okta, but the customers Okta supported and had access to.**
- **Okta's implementation of Zero Trust architectures in question given level of access available to third-party support engineer account.**
- **Okta estimates at least 366 unique clients' organizational data was likely accessed by the threat group via the initial compromised privileged account.**

Over the course of this research, DarkOwl analysts have witnessed – in real time – the terrifying realization of the dangers to software supply chains via malicious compromise of the tools and infrastructure critical to supporting the software development lifecycle.

NOTE: Our analysts concentrated their research on threats to Github, with limited review of Bitbucket and Gitlab vulnerabilities. AWS CodeCommit, IBM Rational, Beanstalk, Azure's Terraform Cloud are additional VCSs for future consideration.

Version Control System Overview

Github

Version control systems are software systems deployed to help developers and their organizations' track code changes to software developed over time. Version control processes and their associated systems are critical in the software development lifecycle.

One well-known example of a free open-source, web-based distributed version control system is Git, which was developed by the author of the Linux kernel, Linus Torvalds.[5] The Git architecture was first deployed in 2005 and now widely used by commercial and personal software developers around the world.

The Git environment facilitates multiple developers simultaneously working on source code software development projects and maintains a record of the changes to source code from initial release to years after deployment. Source code and their associated libraries and executables are stored in repositories, often concatenated to simply, 'repos'.

Github is a cloud-based hosting service that organizes Git repositories via a web interface. Github is a for-profit organization, launched in 2008, and developed on the premise of providing the architecture, community resources, and project management utilities of Git repositories for successful software deployment. Microsoft completed the acquisition of Github from its founders in 2018 for $7.5 billion USD.[6]

Github is built on Git and colloquially, the terms Git and Github are often used interchangeably, but their purpose and functions are separate and unique.

Gitlab is another for-profit company, founded in 2011, that offers similar services to Github. The main difference is Gitlab's implementation of built-in continuous integration/continuous delivery (CI/CD) which helps streamlines the processes between developer code-changes, automated building, testing, and validating updates, and deploying the new improvements to the customer. Gitlab attempts to automate as much of this process, facilitated custom workflows, and reduce need for human involvement in DevOps.

Github has its own version of the CI/CD pipeline through a utility called, "Github Actions."

VCS Used to Distribute Malware Directly

Since Ukraine was invaded in February 2022, DarkOwl has observed a surge the availability of malicious repositories stored in Git and distributed via Github to fuel the hacktivist movement and cyber offensive campaigns against Russia and Belarus.

DDoS attack scripts, malware executables, network targeting data, and leaked databases have all appeared on the platform in the last month.

This surge is not surprising, as Github has become nearly as popular with criminals for storing and sharing malware as it is with software developers uploading legitimate software projects. In recent years, DarkOwl has observed threat actors share links directly to their malicious software repos, often used by organized financial crime and ransomware gangs across numerous darknet malware forums and discussion groups.

Because the platform lacks code purpose monitoring, all types of malware and viruses are readily available on Github including information stealers, password crackers, and botnets. Security researchers discovered Bitbucket is also home to several critical malware distributions including information stealers like **Predator** and **Azorult**. A campaign last year using Bitbucket repositories resulted in over 500,000 victims compromised.[7]

Many threat actors conceal the lethality of their repos by labeling the code with caveats like “For educational purposes only” overtly claiming the code should not be used for illegal gain in descriptions shared on the platform, but state otherwise in associated darknet discussion forums. Other malware groups change the account username associated with the malware frequently to maintain availability.

Last fall, a new strain of malware researchers called “Gitpaste-12” sparked concern as it was the first-ever “worming botnet” exploiting 12-different attack modules. The Gitpaste-12 botnet hosted its main shell script directly on Github and recursively called the malware across Github and Pastebin.

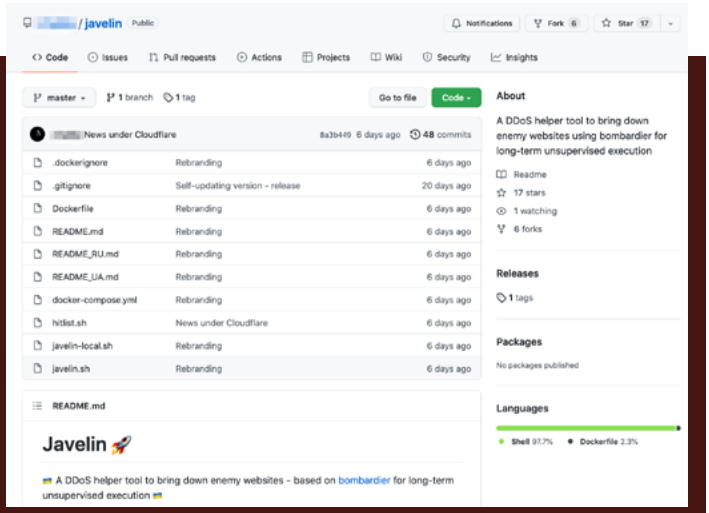


Figure 1 Source redacted for security purposes

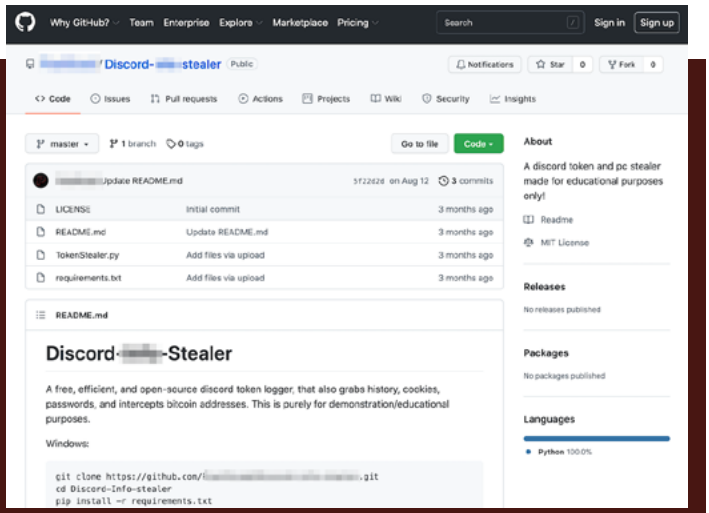


Figure 2 Example Github Repository Containing Malware labeled “For Educational Purposes Only” Source Redacted

According to security researchers investigating the botnet, the shadu1 script in the malware included comments in the Chinese language indicating its possible country of origin – and included commands to disable cloud security agents.[8] The malware targeted Linux servers with x86 architecture and Internet of Things (IoT) devices based on ARM and MIPS chips.

Github immediately shutdown the malicious repository upon discovery.[9]

As of time of writing, DarkOwl has over 9.3 Million documents containing links to software shared on Github. This number has exponentially increased in recent weeks, with over 700,000 documents crawled in the last 6 months. Many of the most recent accounts archived in DarkOwl Vision were manually collected for an investigation or client-specified research.

The example provided below includes a discussion between threat actors on a popular darknet forum explaining how to use Cobalt Strike C2 botnet executables shared on Github.

DETECTING GITHUB LINKS IN VISION

DarkOwl's crawlers do not automatically follow github.com outlinks as Google actively scrapes the service and indexes the content directly for OSINT searches via the Google search engine. However, several links are shared in darknet forums, marketplaces, and public channels. DarkOwl Vision users can employ a simple regular expression (RegEX) to display documents containing content containing a URL with the github.com domain.

```
/(https?\:\/\/)?(www\.)?github\.com.*/
```

[redacted]
Crawled on 2021-08-16 07:50:52 PM

а пока расскажу как это запустить .. заходим из консоли в папку с файлами Cobalt strike 4
вбиваем права на исполнительные файлы

chmod 775 teamserver.sh
chmod 775 cobaltstrike.sh
chmod 775 cobaltstrike.jar

./teamserver.sh ip password запускаем тим сервер с вашим ип вашего сервера и вашим паролем ..
строчка должна например выглядеть так ./teamserver.sh 192.168.1.33 qwerty

запускается ГУИ клиента
из под линукса ./cobaltstrike.sh а из клиент запускаем из под виндовс то cobaltstrike.bat

далее вылетает ГУИ интерфейс в котором задаем ИП тимсервера 192.168.1.33 любой логин например NEO... + пароль что вбили при запуске тимсервера qwerty в нашем примере
.....
а вот далее уже у меня начинаются вопросы

1 какой смысл в профилях ? как например эти https://github.com/[redacted]Malleable-C2-Profiles
2 как просканировать диапазон ип ? я так и не понял либо у меня плохая нпиратка или что ..но при задание сканирования просит выбрать беекон тоесть сессию уже скомпрометированного компютера ...хмм что за бред ? ... я просто хочю из кобальт 4 просканировать диапазон ещо не скомпрометированных машин ...
как это сделать ??? или как то импортировать в кобальт результат с которым можно так же в гуи поработать как это было ранее в кобальте или том же армитаже ???

3 где список эксплоитов в меню ? я так понял теперь самому надо искать по сети скрипты написаным спецом под Cobalt strike 4 эксплоитов ?

Figure 3 Darknet users discuss how to use a repo containing Cobalt Strike Malleable-C2-Profiles and share the Github links with the forum users. DarkOwl Vision DocID: d7e2f640ca56a419f3cdacabddc5d6b4e22b9965, Github source redacted

VCS Used to Distribute Malware Indirectly

DarkOwl has observed Github indirectly exploited via supply-chain style attacks to distribute malicious software to innocent software users who rely on dynamically linked libraries hosted on the VCS. Similar parallel threat vectors have been described as "repo jacking" and "dependency poisoning" in underground communities and both are discussed in more detail below.

Threat actors target vendor software and popular build executables distributed via Github and piggyback on legitimate software as their malware-delivery mechanism. Some attack methods and corresponding darknet intelligence, when available are outlined in the sections below.

SHASUM VIA BASH UPLOADER

In Spring 2021, Codecov alerted customers that their Bash uploader had been compromised via "periodic, unauthorized access to a Google Cloud Storage (GCS) key" earlier in the year, which allowed a threat actor to upload a malicious version of their bash uploader script to the Codecov Github continuous integration (CI) repository used by clients. A customer detected the corrupted software executable on Github after performing a checksum validation against the Github version of shasum (Codecov's coverage reporter) against the version hosted by Codecov directly and discovered discrepancies.[10]

WITH INCREASED POPULARITY IN CLOUD-BASED CODE STORAGE SYSTEMS, THREAT ACTORS HAVE DISCOVERED THEY CAN INFECT PUBLIC GIT AND BITBUCKET REPOSITORIES TO DISTRIBUTE MALWARE, MANY TIMES IN THE FORM OF EMBEDDED BROWSER INFORMATION STEALERS, PERSISTENT BACKDOORS, AND CRYPTO MINERS.

According to their publicized security report, the altered version of the script harvested sensitive information listed below and recommended all users revoke their credentials to mitigate future targeting:

- **Any credentials, tokens, or keys that customers were passing through their CI runner that would be accessible when the bash uploader script was executed.**
- **Any services, datastores, and application code that could be accessed with these credentials, tokens, or keys.**
- **The git remote information (URL of the source) of repositories using the bash uploader to upload coverage to Codecov in CI.**

It is unclear from open-source reporting how many customers were affected by this nefarious supply-chain attack; one that went undetected for months. Codecov has over 29,000 enterprise customers, including high value clients such as: GoDaddy, Washington Post, Royal Bank of Canada, and Procter & Gamble.[11]

Darknet threat actors regularly sell compromised Google Cloud authentication keys on darknet and deep web malware forums and marketplaces. Historical Russian-language darknet discussion groups include several threads detailing how to easily compromise Google accounts and extract cloud storage keys through social engineering and secret question exploitation.

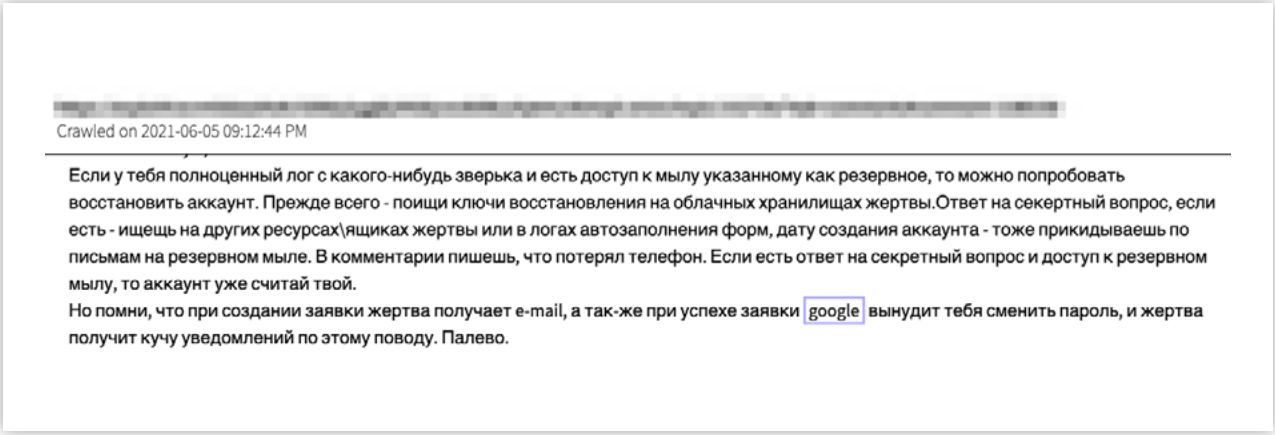


Figure 4 DarkOwl Vision DocID: 07f9b42428b6937b2433044420f85d3edc3eab8b

[FIGURE TRANSLATED]

If you have a full-fledged log from some (victim) and have access to the (file) specified as a backup, then you can try to restore your account. First of all, look for recovery keys on the victim's cloud storage. The answer to the secret question, if there is - search on other resources \ victim's mailboxes or in the autofill logs of forms, the date of account creation - you also estimate by letters on the backup soap. In the comments you write that you lost your phone. If there is an answer to the secret question and access to the backup (file), then the account is already yours.

But remember that when creating an application, the victim receives an e-mail, as well as if the application is successful, Google will force you to change your password, and the victim will receive a bunch of notifications about this. Pale yellow.

NPM

RIAEvangelist's infection of Node.js related NPM packages was not the first observed instance of NPM poisoning. In early 2021, Github detected a widespread cryptocurrency mining campaign that infected thousands of repos across their servers by circulating a malicious version of NPM's executable, npm.exe.[12]

NPM has been regularly exploited by malicious actors. For the average JavaScript developer, 90,000 NPM packages are downloaded annually making it near impossible to detect an infected NPM package.

Last fall, security researchers detected the popular UA-Parser-JS NPM library was similarly infected and altered to steal passwords along with critical browser identification parameters: browser user agent, search engine, OS, CPU, and the victim's device type and model. This library is critical to JavaScript developers using Linux and MacOS. According to the library developer, Faisal Salman, the package was commandeered after his personal NPM account was compromised. The package is downloaded an estimated eight million times a week and is used by tech giants such as: Google, Amazon, Facebook, IBM, and Microsoft.[13, 14]

The topic of NPM hijacking has been widely discussed in prominent darknet malware discussion forums. In November 2020 (shortly before the surge in NPM related supply-chain infections in early 2021), a darknet forum user started a thread titled, **"Статья NPM Hijacking. Встраиваем произвольный код в приложения на Node.js"** [Translated: "NPM Hijacking article. Embed arbitrary code in applications on Node.js"]. The forum post detailed the pervasive use of Node.js and how NPM is a worthy attack vector for embedding malware. The techniques described in the user's thread was analogous with DLL hijacking.

> **NODE.JS ПОВСЮДУ** [Translated: "NODE.JS IS EVERYWHERE"] (in all caps for emphasis) – posted by user tobacco on a darknet forum.

The user details the process how to corrupt the original library by linking to a malicious clone of the library:

> Если разработчик не указал явным образом пути для загрузки библиотек DLL файла в своем исполняемом файле, то операционная система будет искать эти библиотеки по путям, перечисленным в переменной пути (в в том числе для Windows, подробнее — в официальной документации). Подложив в какую-то из этих папок вредоносный клон библиотеки, атакующий может исполнить произвольный код.

[FIGURE TRANSLATED]

If the developer has not explicitly specified the paths for loading the DLL file libraries in his executable file, then the operating system will search for these libraries along the paths listed in the path variable (including for Windows, for more details, see the official documentation). By placing a malicious clone of the library in one of these folders, an attacker can execute arbitrary code.

The lengthy, but educational post continues:[15]

> И хотя NPM Hijacking — это по большей части то, что называется security weakness, такая атака может быть прекрасным элементом в цепочке эксплуатации. Она позволяет выполнять код от имени подписанных исполняемых файлов, причем в некоторых случаях с более высокими правами, чем у текущего пользователя. Либо вредонос может попытаться обмануть пользователя, так как запрос от UAC на повышение прав поступит от привычной ему программы.

> Итого NPM Hijacking — неплохой инструмент в арсенале Red Team, потому что это: просто и стабильно; кросс-платформенно; можно назвать некой «ленивой» альтернативой Meterpreter; может не отлавливаться детекторами аномалий, которые ждут подвоха скорее от обращения к сторонним веб-ресурсам из Word, какого-нибудь PDF Reader или выполнения команд через cmd или PowerShell. Здесь же код выполняется в рамках конкретной программы, которая и так работает с разными ресурсами

[FIGURE TRANSLATED]

> And although NPM Hijacking is for the most part what is called security weakness, such an attack can be a wonderful element in the chain of exploitation. It allows you to execute code on behalf of signed executable files, and in some cases with higher rights than the current user. Or the malware may try to deceive the user, since the request from the UAC to increase the rights will come from a program familiar to him.

> In total, NPM Hijacking is a good tool in the arsenal of Red Team, because it: simple and stable; cross-platform; you can call it a kind of “lazy” alternative to Meterpreter; it may not be caught by anomaly detectors, which are waiting for a trick rather from accessing third-party web resources from Word, some PDF Reader, or executing commands via cmd or PowerShell.

PHP

Infected NPM attack methodologies are nearly identical to how threat actors compromised the central PHP library in Spring 2021. An unknown threat actor compromised the official PHP Git repository with “minor edits” that created a backdoor for malicious remote code execution on victim servers. Security researchers discovered the attacker uploaded the malicious library to **git.php.net** and the PHP repositories on Github directly infecting an unknown number of users who inadvertently downloaded the corrupted binaries.[16]

Even before discovering malicious codes embedded in NPM and PHP, Github representatives have publicly stated they assess an increased risk of compromise to many other critical packages including **Composer, Maven, NuGet, PyPI, and RubyGems**.[17]

PYPI

In August 2021, a Japanese researcher known simply as **RyotaK** discovered critical vulnerabilities in the Python Package Index (PyPI) that could arbitrarily execute code and take full control of the official software repository – including deleting project documentation files.[18] The vulnerabilities reportedly exploit Github Actions, which is discussed in more detail below.

The researcher detailed additional concerns:

> "A more critical flaw concerns an issue in the **Github Actions** workflow for PyPI's source repository named "combine-prs.yml," resulting in a scenario wherein an adversary could obtain write permission for the main branch of the "pypa/warehouse" repository, and in the process execute malicious code on pypi.org."

Github has supposedly mitigated these vulnerabilities with Github Program Manager, Chris Patterson stating:

"A pull request from the first contributor requires manual approval from a repository collaborator with write access before the action workflow is executed. When the first contributor opens a pull request, the admin will need to approve the action workflow and then a message will appear stating that the workflow can be run after that."

The researcher RyotaK is mentioned several times in darknet forums for their contributions, including vulnerabilities they discovered in Homebrew, cdnjs and Nvidia's GE Force. Their blog: **blog.ryotak.me** links to their Github and Twitter accounts and documents many of their research findings, including detailing how Github Actions can be compromised by a threat actor stealing the GITHUB_TOKEN.[19]

Additional Surreptitious Threat Attack Vectors

GITHUB ACTIONS EXPLOIT

Vulnerabilities within Github's CI/CD utility, 'Github Actions' has been a popular threat attack vector against Github infrastructure for quite some time. Over the last year, Github users have had their programs hijacked and malicious crypto-mining codes launched through simple pull requests via the CI/CD system.[20]

Most attacks involve creating a fork of a legitimate repository, adding malicious GitHub Actions to the source, and then sending a Pull Request from the original repository to merge the code back with the original.[21]

In October 2021, researchers from Cidr Security surfaced a security flaw within Github Actions and system permissions of GITHUB_TOKEN that allowed a malicious threat actor to push code into production without review or supervisory approval of a protected branch.[22]

In March 2022, another security company, Cycode identified three major risks in Github Actions in the workflow queuing model and runner environment:[23]

3 MAJOR RISKS IN GITHUB ACTIONS

1. **A malicious threat actor can cause a critical supply chain incident by committing undesired code that introduces backdoors deployed to end-users or organizational environments.**

   This is achieved by an attacker fetching sensitive tokens (GITHUB_TOKEN or a personal access token) with write permissions for that repository.

2. **A malicious threat actor could exfiltrate workflow secrets and, in some cases, source code, repository or organization secrets.**

   Example secrets include tokens for private repositories, container registries, cloud assets, or any other sensitive information.

3. **A much smaller risk would be the malicious actor's ability to run botnets or crypto miners using the runner infrastructure.**

In May 2021, DarkOwl observed Russian malware developers on the AntiChat Telegram channel recommending developers run Github Actions locally to test malicious code injections before the official push and/or pull to the target repository.

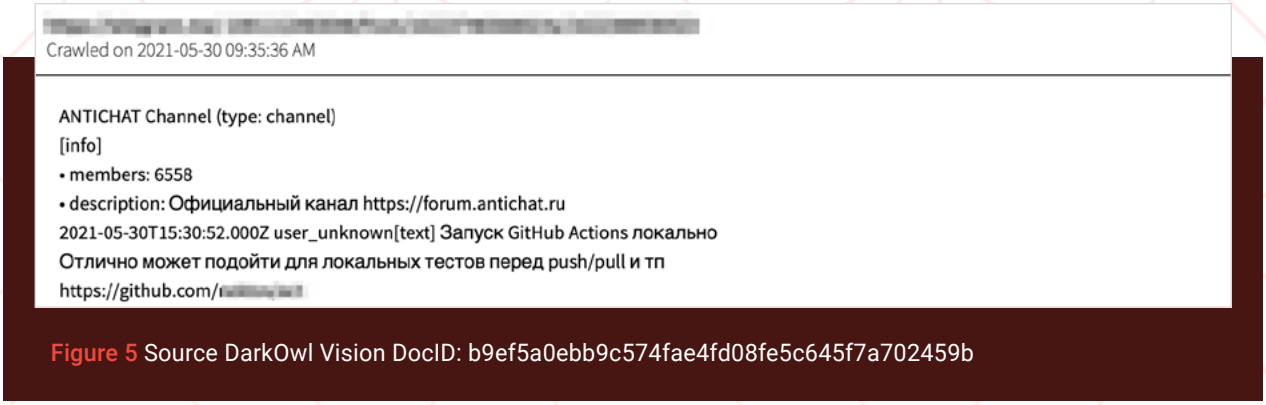


Figure 5 Source DarkOwl Vision DocID: b9ef5a0ebb9c574fae4fd08fe5c645f7a702459b

[FIGURE TRANSLATED]

description: Official channel https://forum.antichat.ru
2021-05-30T15:30:52.000Z user_unknown[text] Running GitHub Actions Locally
It can be perfectly suitable for local tests before push/pull, etc

OCTOPUS SCANNER

Last Spring, Github discovered it was the source of a major supply chain attack against the Java integrated development environment (IDE), Netbeans with malware payloads originating in February 2019. Dubbed **Octopus Scanner,** the malware infected the build process such that every time a project was built any resulting JAR files were infected with the "dropper" to execute the malicious instructions.

Once installed locally, a remote access trojan (RAT) activates and reports back the name, architecture, username of the operating system, and username of the .gitconfig file back to dedicated malicious C2 IP addresses for further exploitation.[24]

After a lengthy investigation, researchers determined at least 26 Github code repositories were affected with the malicious codes with an unknown number of downloads to infected developer machines.[25]

REPOSITORY HIJACKING

DarkOwl has observed threat actors conduct direct Github repository hijacking (a.k.a. "repo jacking") by multiple means. One simple method is through exploiting username changes. When a linked repository account holder changes their username, their repositories become immediately available for re-registration by any user. Similarly, when a Github user transfers their repository to another user or organization and then deletes their account, the redirect is vulnerable to hijacking.[26]

Developers who link to legacy code repositories that have changed ownership are at risk of their code having dependencies to possible malicious versions of the original repositories.

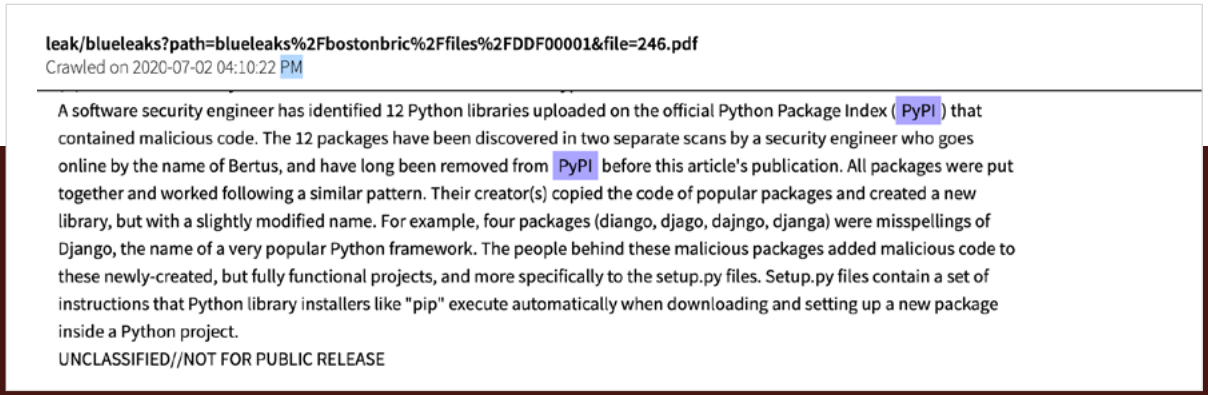
One open-source resource that helps developers visualize potential dangers in code dependencies is Libraries.io, which enables monitoring millions of open-source packages across different package managers, which helps to mitigate the risk of repo-jacking.[27] Researchers also advise not to link directly to Github repositories as they are not static, nor a suitable replacement for an official software package manger.

DEPENDENCY POISONING

A common method for dependency poisoning (a.k.a. dependency confusion) is typo-squatting package names. In the same way that malicious URLs are specifically created redirect visitors to a phishing or malicious version of a legitimate website, threat actors are now deliberately establishing malicious copies of existing packages with miniscule spelling changes or simple replacements of a capital "O" with the number "0" (for case-sensitive packages). The intention is that developers inadvertently link to the malicious library instead of the legitimate one.

Security researchers detailed how a popular Golang package **loqrus** was exploited when two letters of one of its imported paths were exchanged: **siruspen** instead of **sirupsen.** To this day, the package is still on Github, and several repositories appear to call the poisoned package.[28]

In the Blueleaks files – a 2020 leak of hundreds of gigs of internal files used by U.S. law enforcement and police departments – numerous similarly typosquatted Python packages were identified to contain malicious code. Four packages were misspellings of Django python framework (diango, djago, dajngo, djanga), preying on non-observant and non-English speaking developers.


leak/blueleaks?path=blueleaks%2Fbostonbric%2Ffiles%2FDDF00001&file=246.pdf
Crawled on 2020-07-02 04:10:22 PM

A software security engineer has identified 12 Python libraries uploaded on the official Python Package Index (PyPI) that contained malicious code. The 12 packages have been discovered in two separate scans by a security engineer who goes online by the name of Bertus, and have long been removed from PyPI before this article's publication. All packages were put together and worked following a similar pattern. Their creator(s) copied the code of popular packages and created a new library, but with a slightly modified name. For example, four packages (diango, djago, dajngo, djanga) were misspellings of Django, the name of a very popular Python framework. The people behind these malicious packages added malicious code to these newly-created, but fully functional projects, and more specifically to the setup.py files. Setup.py files contain a set of instructions that Python library installers like "pip" execute automatically when downloading and setting up a new package inside a Python project.

UNCLASSIFIED//NOT FOR PUBLIC RELEASE

Figure 6 DarkOwl Vision DocID: f15bd37c8fa87bc6993b62f71e725954ee88afee

On the popular Chinese Software Development Network, known in use by Chinese state-sponsored threat actors, one user account (smellycat000) detailed the success of typosquatting Python multiple libraries through the PyPI portal. The user wrote:

[Original Text in Chinese]

> **这些程序包可被滥用于成为更复　威胁的入口点，使得攻击者能够在目标机器上执行　程代码、收集系统信息、窃取信用卡信息和自动存储在 CHROME 和EDGE 浏览器中的密码，甚至窃取 DISCORD 认证令牌假冒受害者。**

[FIGURE TRANSLATED]

These packages can be abused to become entry points for more complex threats, enabling attackers to execute remote code on the target machine, collect system information, steal credit card information and passwords automatically stored in Chrome and Edge browsers, and even steal Discord authentication tokens to impersonate victims.

Figure 7 Source: https://blog.csdn.net/smellycat000/article/details/119337824?spm=1001.2014.3001.5501

RANSOMING REPOSITORIES

A couple of years ago, security researchers detected a trend across Bitbucket, Github, and Gitlab where threat actors were breaking into repositories, wiping the data, and demanding a cryptocurrency ransom for the data's return.[29] According to documents in DarkOwl Vision, the extortionists used the e-mail address **admin@gitsbackup.com** to communicate with their victims.

Github's Git Ransom Incident Report revealed that the threat actors scanned their servers for publicly exposed '.git/config' files containing sensitive data, such as credentials and personal tokens that were utilized in an automated credential stuffing campaign to perform Git push commands to issue the ransom note and take control of the accounts.[30] Other victims claimed they had reused previously exposed passwords on their Github accounts and others had their .DS_Store file stored on Mac-based operating systems compromised.

Some more sophisticated threat actors will brute force their way into a private code repository using exposed credentials or password crackers, such as what happened when prominent darknet data broker, ShinyHunters compromised Microsoft's "Private" Github account in May last year.[31]

SINCE AUGUST 2021, GITHUB HIGHLY ENCOURAGES ALL ITS USERS TO ENABLE MULTI-FACTOR AUTHENTICATION OR EVEN SETUP PHYSICAL DIGITAL SECURITY KEYS (E.G. YUBIKEYS) TO REDUCE THE RISK OF ACCOUNT TAKEOVER.[32]

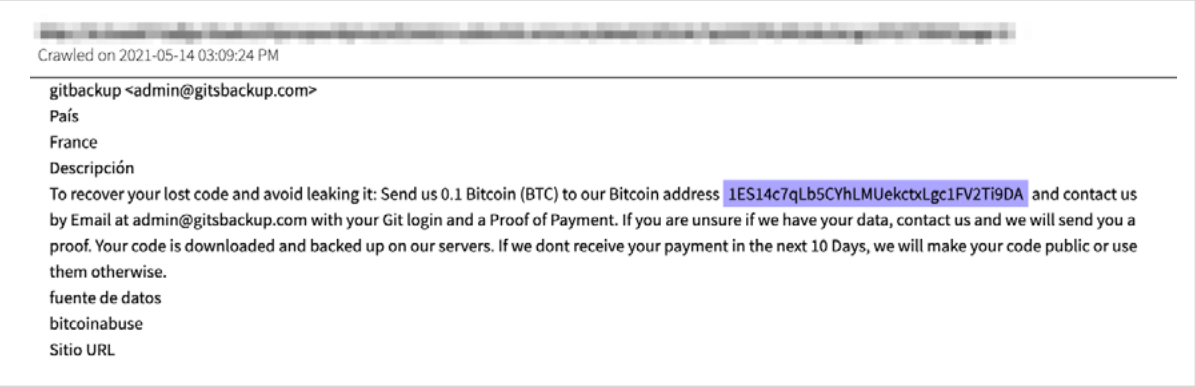


Figure 8 DarkOwl Vision DocID: 794cebdbc17c98ffa38317f557dbbd75d798c1c7

Widespread Supply Chain Risks Persist

In recent years, the compromise of Solarwinds and Kaseya customers via malicious software upgrades and poisoned software distribution channels highlighted the need for widespread zero-trust architecture adoption. Organizations around the US and western Europe rallied around this philosophy and adopted new security applications and technologies to reduce their security risk, i.e. digital identity protection, secured containers for critical data, etc.

LAPSUS$ – an emergent darknet cyber threat actor that recently gained notoriety for leaking source code from Samsung, Nvidia, Microsoft, and other high valued targets – has now weaponized software supply chains for even more significant impact.

Recent revelations that LAPSUS$ successfully infiltrated Okta, a digital identity verification and authentication platform, in late January, and subsequently accessed sensitive data for at least 360 additional victims via Okta's client interface, brings into question the concepts of digital trust and zero-trust mandates at the enterprise level. The disclosure of the Okta breach and subsequent client compromises was not public until LAPSUS$ shared screenshots as proof of their privileged access to Okta on Telegram in March 2022.

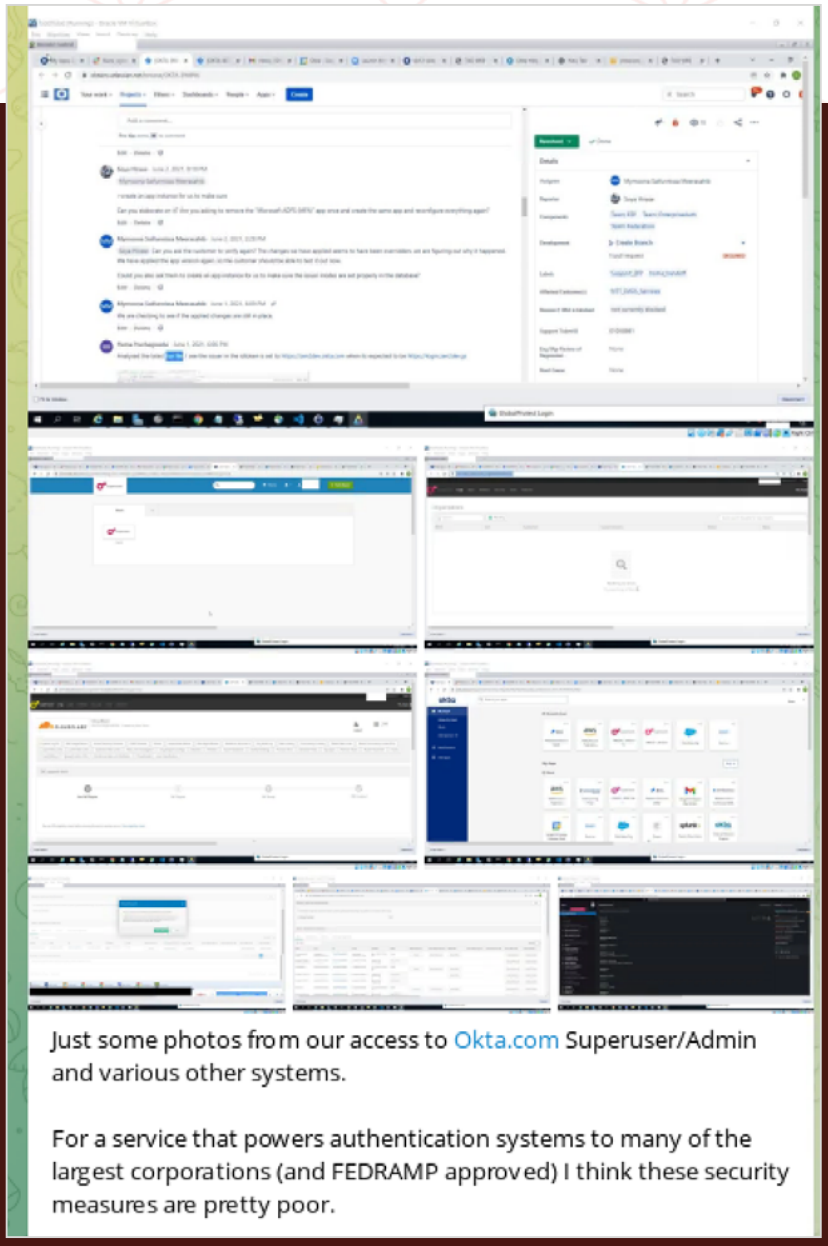


Figure 9 Screenshots Released By LAPSUS$ Proving Access to okta.com's Internal Systems

According to LAPSUS$'s Telegram channel and open-source reporting, the initial point of entry into Okta was a compromised third-party support engineer's thin client. LAPSUS$ has confided they have used previously utilized darknet markets, such as Genesis Market to purchase stolen cookies and personal authentication data to employ in offensive cyber operations.

While there is no certainty that LAPSUS$ purchased the Okta's employee credential data via Genesis, DarkOwl collects listings offered on the Genesis Market and has found hundreds of botnets and browser stealer logs on offer that include sensitive okta.com session data. For less than $100 USD, a malicious threat actor can acquire the login data and personal authentication tokens for Okta in addition to other cookie data for services like Payoneer, Atlassian, and Paypal.

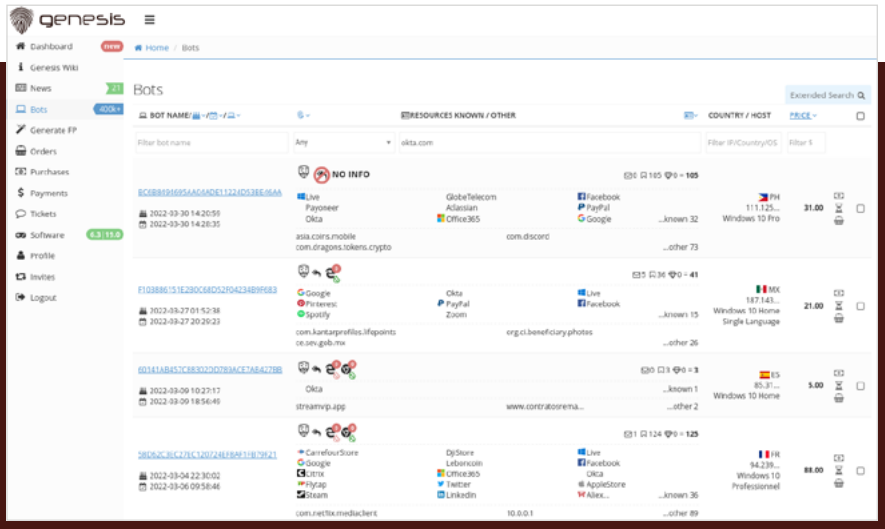


Figure 10 Genesis Market Botnet Search Query with Okta Mentions; Over 400,000 Compromised Terminals Available to Purchase. Source URL Redacted for Security

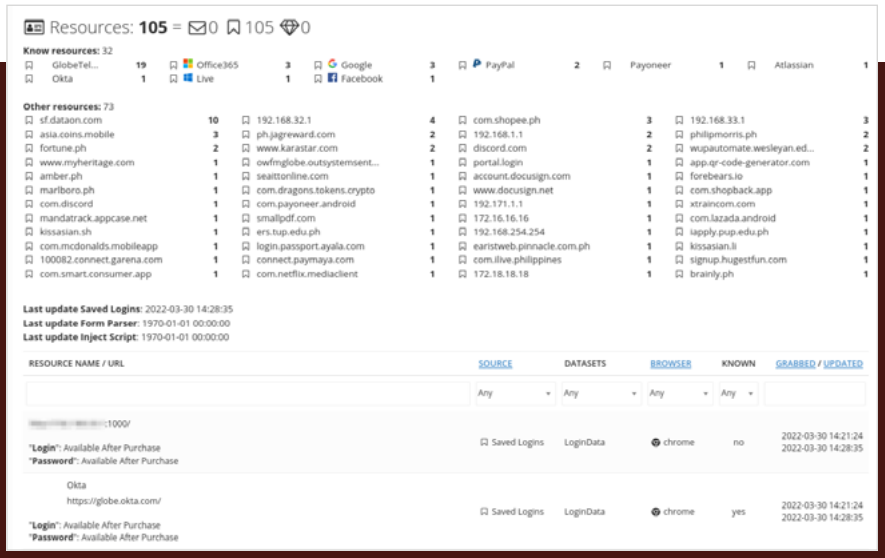


Figure 11 Botnet Data for Sale Which Includes 105 Cookies and Login Credentials for Various Websites

According to open-sources,[33] once LAPSUS$ successfully gained access to Okta's internal resources via the Remote Desktop Protocol (RDP), they escalated their stolen user's privileges using repositories they downloaded via Github. After terminating software security endpoints, like FireEye, they downloaded Mimikatz from Github and used the exploitation utility to extract credentials from memory, hashes, PINs and Kerberos tickets.[34] The LAPSUS$ group taunted Okta's public response by claiming that the support engineer had access to over 8,600 internal Slack channels where they discovered employee AWS credentials and Github private tokens stored in plaintext. The volume of access for a third party vendor or contractor is contradictory to the fundamentals of zero/ limited trust implementations.

Less than a month ago, LAPSUS$ released to their Telegram channel a 70GB torrent of data exfiltrated from Globant along with the administrator credentials for Globant 's Confluence, Jira, and Github instances. Globant is an international IT and Software Development consultancy company with over 25,000 employees and supports thousands of clients, such as C-SPAN, Facebook, DHL, Abbot, and Stifel.

It is abundantly clear the supply chain infections will continue, and malicious threat actors will exploit software supply utilities and toolsets to their benefit. **While LAPSUS$ – believed to be at least in some part a mixture of self-taught teenagers in the UK and Brazil with too much time on their hands and motivated by the desire for clout and public infamy – easily infiltrated and exfiltrated hundreds of gigabytes of sensitive data, highly sophisticated nation state cyber actors could equally easily infect supply chains to an even worse extent, establishing persistent access for espionage and triggering destructive critical infrastructure attacks.**

VERSION CONTROL SYSTEMS AND SOFTWARE SUPPLY CHAINS ARE A VIABLE AND HIGH CONSEQUENTIAL ATTACK VECTOR READILY EXPLOITED BY CYBERCRIMINAL ORGANIZATIONS, NATION STATE ACTORS, AND HACKTIVISTS FROM THE DARKNET. DARKOWL BELIEVES THERE WILL BE CONTINUED AND INCREASED ATTACKS AGAINST DEPENDENCY LIBRARIES AND SOFTWARE PACKAGE MANAGERS, SUCH AS NPM AND PYPI, WITH THE INTENTION OF STEALING INFORMATION AND ESTABLISHING LONG TERM PERSISTENCE IN THE VICTIM MACHINES.

Security Paths Forward

VCS RECOMMENDATIONS FOR USERS

Academic researchers have been studying how to detect malicious programs on VCS's like Github. In a collaborative publication between scientists at Github, its parent Microsoft, and the Rochester Institute of Technology earlier this year, the authors introduce a platform called, "Anomalicious" designed and developed to "automatically detect and flag anomalous and potentially malicious commits" to the Github VCS using commit logs and repository metadata. This is likely a technological security path forward for Github to detect and remove malicious repositories directly hosted on its platform.

Gitlab advocates for signing code all commits used in the software build process "early and often." By implementing organizational-wide code signing that involves cryptographic hashing, the integrity of the developer's code and authenticity of the software package is maintained to prevent tampering since being published.

Github published security and compliance policy recommendations for users of Github Actions to mitigate security risks. Their recommendations included:

Regularly auditing access to key repositories;

Enforcing security policies;

Demonstrating traceability with issue branches; and

Demonstrating traceability by requiring every pull request to have a corresponding link issue.

SOFTWARE SUPPLY CHAIN SECURITY

The increasingly terrifying level of access and exfiltration occurring by threat actors such as LAPSUS$ exploiting the very essence of software security and risk via digital authenticators, presents a bigger strategic challenge of how-to-best secure the software development lifecycle.

Every organization must closely evaluate each and every utility in use by its corporation and its employees, including but not limited to version control system, cloud providers, and digital authenticators. Personal security measures need reconsideration and implement better security hygiene, e.g. not storing credentials and login data in the browser, especially in a hybrid work-from-home development environment.

Storing AWS security keys and Github PRIVATE TOKENS into cleartext Slack channels or exporting LastPass credential data into Excel spreadsheets cannot continue. At the same time, CISOs and security strategists must recognize rigorous security measures are possibly burdening productivity and leading to gross security lapses in the process.

Furthermore, the corporate adoption of a password storage manager, such as LastPass has historically been the recommended solution for preventing unauthorized domain access via browser stealer botnet exploits; still, it only takes a supply chain attack directed against LastPass to then have millions of credentials exposed and leaked to the darknet, destroying the digital trust placed in such security platforms.

Conclusions

DarkOwl believes code repositories such as Github will continue to be a key resource for hosting malicious software samples and exploits for use in criminal cyber campaigns to target software supply chains.

The research compiled in this paper attempts to covers several noteworthy vulnerabilities in the software supply chain, with special attention to version control systems, but recognizably not every exploit was explicitly discussed. Since our analysts submitted their initial draft, we've observed additional software VCS platform vulnerability disclosures. In late March, Gitlab disclosed a critical security release impacting Community Edition (CE) and Enterprise Edition (EE) that allowed for threat actors to take over Gitlab accounts via hardcoded passwords in OmniAuth. Threat actors could also execute malicious cross-site scripting (XSS) code by injecting malicious HTML into Gitlab notes.[38] A similarly severe release was released in late February for Gitlab after a security researcher discovered an unauthorized user could steal runner registration tokens using quick actions commands.[39] VCSs are improving equally addressing platform security issues. Last week, Github announced new features for their Advanced Security cloud platform users that includes "push protection" to help prevent the accidental exposure of organizational credentials such as API keys and secret tokens.[40]

DARKOWL ASSESSES WITH HIGH CONFIDENCE THAT SOFTWARE SUPPLY-CHAIN ATTACKS EXPLOITING VULNERABILITIES IN POPULAR VERSION CONTROL SYSTEMS AND SOFTWARE PACKAGE MANAGERS WILL CONTINUE WITH INCREASING FREQUENCY AND CONSEQUENCE.

Threat actors have discovered the ease in which poisoned library dependencies can be circulated through such systems with little to no detection or auditing for malicious code. Widely popular and extensively used packages such as NPM and PyPI will continue to be targeted to infect victim devices with information stealers, crypto miners, and persistence-focused malware.

We believe the latest software supply chain compromises like Okta are just the beginning of an ever complex and increasingly lethal cyber-attack vector. Any product or service that touches one's network or supports the Software Development Lifecycle, i.e. Customer Relationship Management software, software version control utilities, identity authenticators, payroll and timekeeping accounting systems, cloud service providers, internal employee messaging platforms (Slack, Teams, etc.) are all potential targets for compromise.

If LAPSUS$ can cause as much chaos as they have in a short time, exploiting a handful of high-valued, prominent corporate victims and their software vendors, how much more damage has a covert, nation-state sponsored threat actor already caused that we are simply unaware of, and anxiously anticipating its inevitable and possibly catastrophic aftermath.

SOURCES

1 https://threatpost.com/dev-sabotages-popular-npm-package-protest-russian-invasion/178972/

2 https://www.csoonline.com/article/3626703/the-kaseya-ransomware-attack-a-timeline.html

3 https://www.nist.gov/publications/zero-trust-architecture

4 https://www.toolbox.com/it-security/security-general/news/lapsus-ubisoft-security-incident/

5 https://linuxtorvalds.com/

6 https://blogs.microsoft.com/blog/2018/10/26/microsoft-completes-github-acquisition/

7 The Bitbucket campaign was originally detected by Cybereason in early 2020.
https://www.cybereason.com/blog/the-hole-in-the-bucket-attackers-abuse-bitbucket-to-deliver-an-arsenal-of-malware

8 https://blogs.juniper.net/en-us/threat-research/gitpaste-12

9 https://threatit.com/articles/gitpaste-12-linux-bot-armed-with-a-dozen-exploits/

10 https://about.codecov.io/security-update/

11 https://www.zdnet.com/article/codecov-breach-impacted-hundreds-of-customer-networks/

12 https://www.bleepingcomputer.com/news/security/github-actions-being-actively-abused-to-mine-cryptocurrency-on-github-servers/

13 https://portswigger.net/daily-swig/popular-npm-package-ua-parser-js-poisoned-with-cryptomining-password-stealing-malware

14 https://blog.sonatype.com/newly-found-npm-malware-mines-cryptocurrency-on-windows-linux-macos-devices

15 Details of this conversation are included on the DarkOwl Vision platform in the DocID: 85e00f605e6c33eae0e7dd27439ed64ba1d6cd3e

16 https://portswigger.net/daily-swig/backdoor-planted-in-php-git-repository-after-server-hack

17 https://www.zdnet.com/article/open-source-software-security-vulnerabilities-exist-for-over-four-years-before-detection-study/

18 https://thehackernews.com/2021/08/pypi-python-package-repository-patches.html

19 https://blog.ryotak.me/post/github-actions-supplychain/

20 A French programmer known as Tib details their experience with malicious Github Actions activity on their account.
https://dev.to/thibaultduponchelle/the-github-action-mining-attack-through-pull-request-2lmc

21 https://threatit.com/articles/attackers-use-the-github-server-infrastructure-for-cryptomining/

22 https://medium.com/cider-sec/bypassing-required-reviews-using-github-actions-6e1b29135cc7

23 https://cycode.com/blog/github-actions-vulnerabilities/

24 https://nsfocusglobal.com/supply-chain-attack-event-targeted-attacks-on-java-projects-in-github/

25 https://securitylab.github.com/research/octopus-scanner-malware-open-source-supply-chain/

26 https://blog.securityinnovation.com/repo-jacking-exploiting-the-dependency-supply-chain

27 https://libraries.io/

28 https://redhuntlabs.com/blog/dependency-confusion-attack-what-why-and-how.html

29 https://www.vice.com/en/article/vb9v33/github-bitbucket-repositories-ransomware

30 DarkOwl archived details of the campaign in the Vision DocID: 26e8cdbdf7e05e9acb7728196c57cb0d79f92761

31 https://www.bleepingcomputer.com/news/security/microsofts-github-account-hacked-private-repositories-stolen/

32 https://github.blog/2021-08-16-securing-your-github-account-two-factor-authentication/

33 https://twitter.com/BillDemirkapi/status/1508527487655067660/photo/1

34 https://www.csoonline.com/article/3353416/what-is-mimikatz-and-how-to-defend-against-this-password-stealing-tool.html

35 https://patricegodefroid.github.io/public_psfiles/icse2021.pdf

36 https://about.gitlab.com/blog/2021/08/30/secure-pipeline-with-single-sign-in/

37 https://github.blog/2021-10-22-github-actions-for-security-compliance/

38 https://about.gitlab.com/releases/2022/03/31/critical-security-release-gitlab-14-9-2-released/#stored-xss-in-notes

39 https://about.gitlab.com/releases/2022/02/25/critical-security-release-gitlab-14-8-2-released/

40 https://www.bleepingcomputer.com/news/security/github-can-now-auto-block-commits-containing-api-keys-auth-tokens/

DARKOWL DATA SOURCES

Tor, I2P, ZeroNet, authenticated forums, darknet marketplaces, IRC, high-risk paste sites, encrypted chat services, and open FTP servers.

ABOUT DARKOWL

DarkOwl uses machine learning to automatically, continuously, and anonymously collect, index and rank darknet, deep web, and high-risk surface net data that allows for simplicity in searching.

Our platform collects and stores data in near real-time, allowing darknet sites that frequently change location and availability, be queried in a safe and secure manner without having to access the darknet itself.

For more information, visit www.darkowl.com