Tenacious Pungsan: A DPRK threat actor linked to Contagious Interview | Datadog Security Labs

By Ian Kretz, Sebastian Obregoso, Datadog Security Research Team

Published: 2024-10-24 · Archived: 2026-04-05 14:54:15 UTC

Key points and observations

- In September 2024, Datadog Security Research discovered three malicious npm packages: `passports-js`, `bcrypts-js`, and `blockscan-api`.
- These packages had a combined 323 downloads and contained samples of BeaverTail malware, a family of JavaScript infostealers and downloaders used by threat actors associated with Democratic People's Republic of Korea (DPRK, also referred to as North Korea).
- Reporting from Palo Alto Networks Unit 42 has associated BeaverTail with an ongoing campaign named Contagious Interview, which targets job-seekers in the US tech industry. Victims are encouraged to participate in a fictitious job interview, during which the BeaverTail malware is delivered as part of an interview task.
- Datadog Security Research has linked the samples presented in this blog to Contagious Interview and attributes them to a single threat actor which we designate "Tenacious Pungsan." (We align nation-state threat actor clusters with their national breeds, and the Pungsan is a dog native to North Korea.)

Background

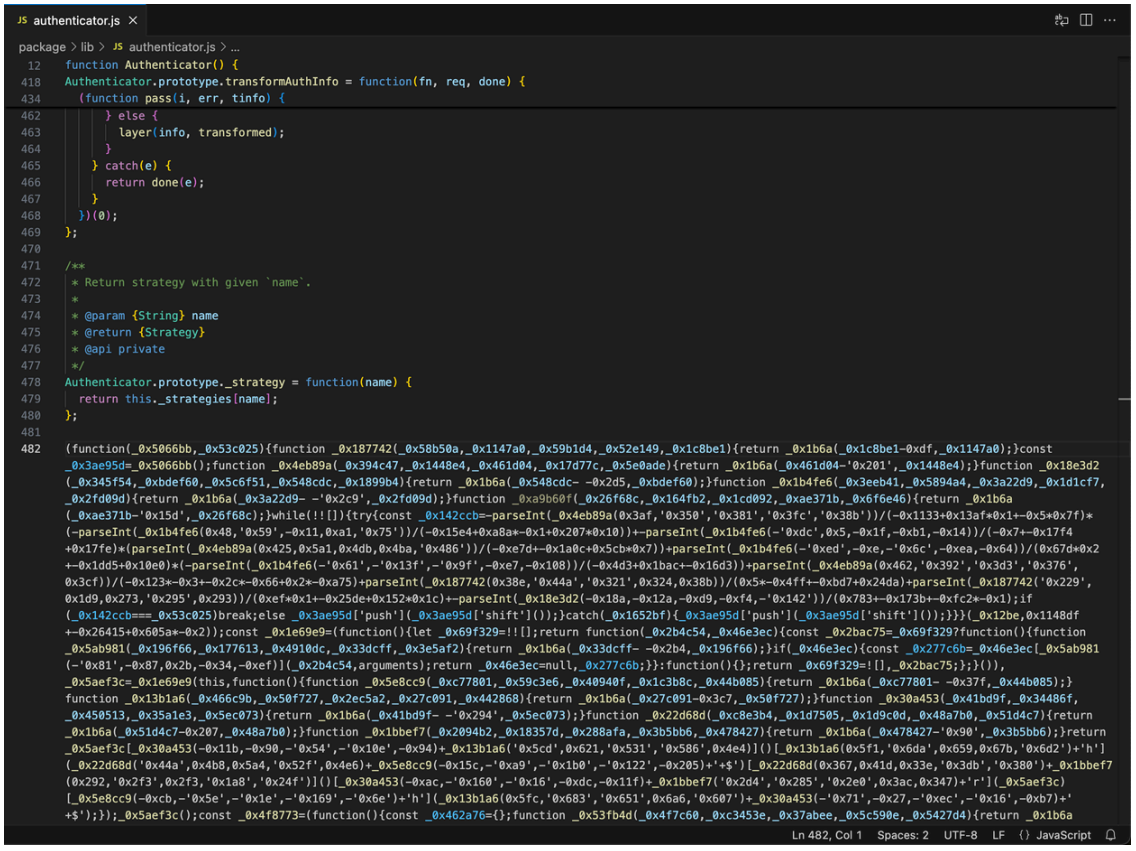
In recent years, the open source software supply chain has become a focus of increasing concern as an effective attack vector for malicious actors to compromise downstream targets. Attackers may seek to compromise existing, often broadly used packages, or they may publish new packages containing malicious code. Attacks of this second kind usually involve some form of namesquatting, in which the name of the malicious package is very similar to a targeted legitimate package in hopes that developers will confuse the former for the latter. We have observed significant attacks of both kinds in 2024 alone.

Datadog Security Research continuously monitors both npm and PyPI for new and ongoing software supply chain attacks. We do so using GuardDog, a command-line scanner for identifying malicious open source packages via code behaviors and package metadata. With the assistance of GuardDog, we have cataloged more than 1,700 (and counting) malicious PyPI and npm packages over the past two years, which we publish in a public dataset.

The timeline

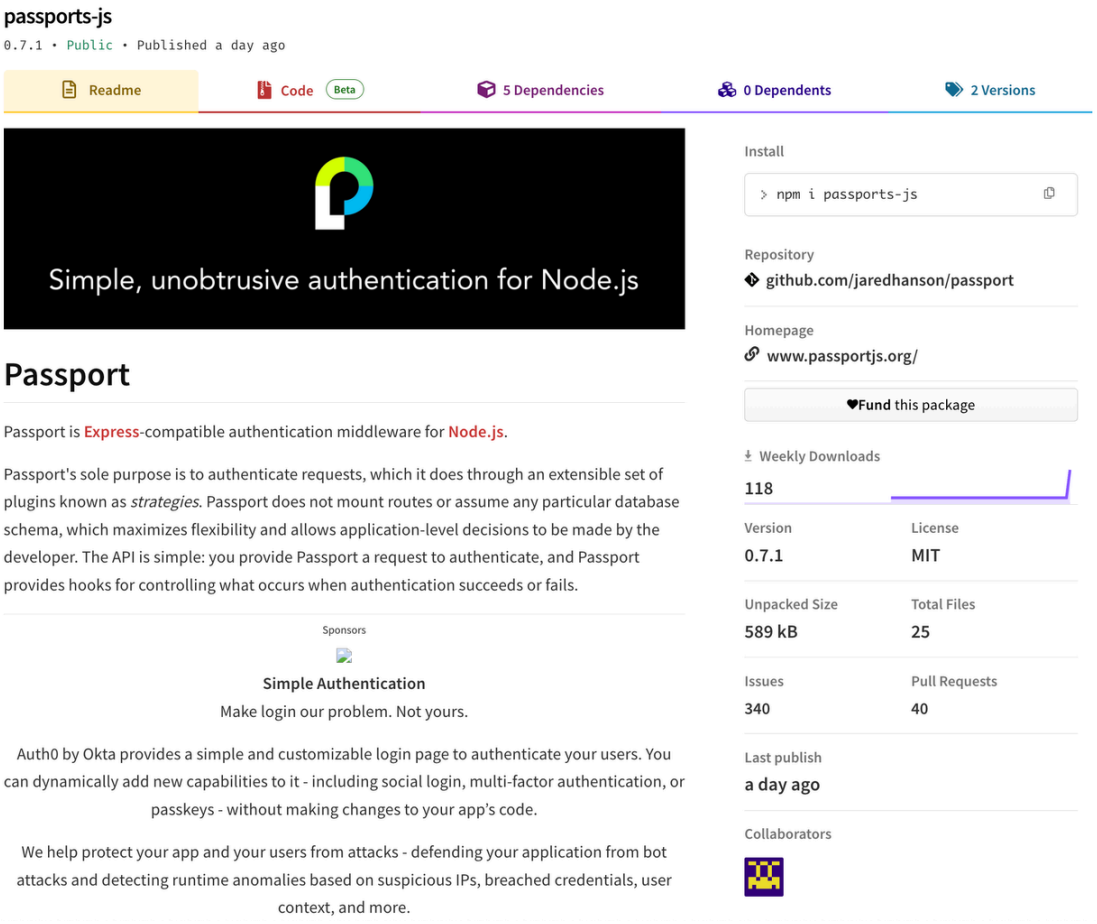
On September 11, 2024, versions 0.7.0 and 0.7.1 of the npm package `passports-js` were automatically flagged for manual triage by a security researcher as part of our continuous monitoring of npm. GuardDog's scans reported that both versions of `passports-js` contained the same very long line of obfuscated JavaScript code in an otherwise unobfuscated source file, giving cause for suspicion.

Code obfuscation is the practice of obscuring the text or behaviors of a unit of code so that they are difficult for humans or automated analyzers to discern. Naturally, it is a routinely deployed tactic in open source malware. Common forms of obfuscation include using random identifiers instead of meaningful ones, removing code formatting, adding useless operations to complicate the code's structure, and concealing code behind nonstandard text encodings or encryption. The obfuscated line found in `passports-js`, shown in the following image, uses all but the last of these techniques.



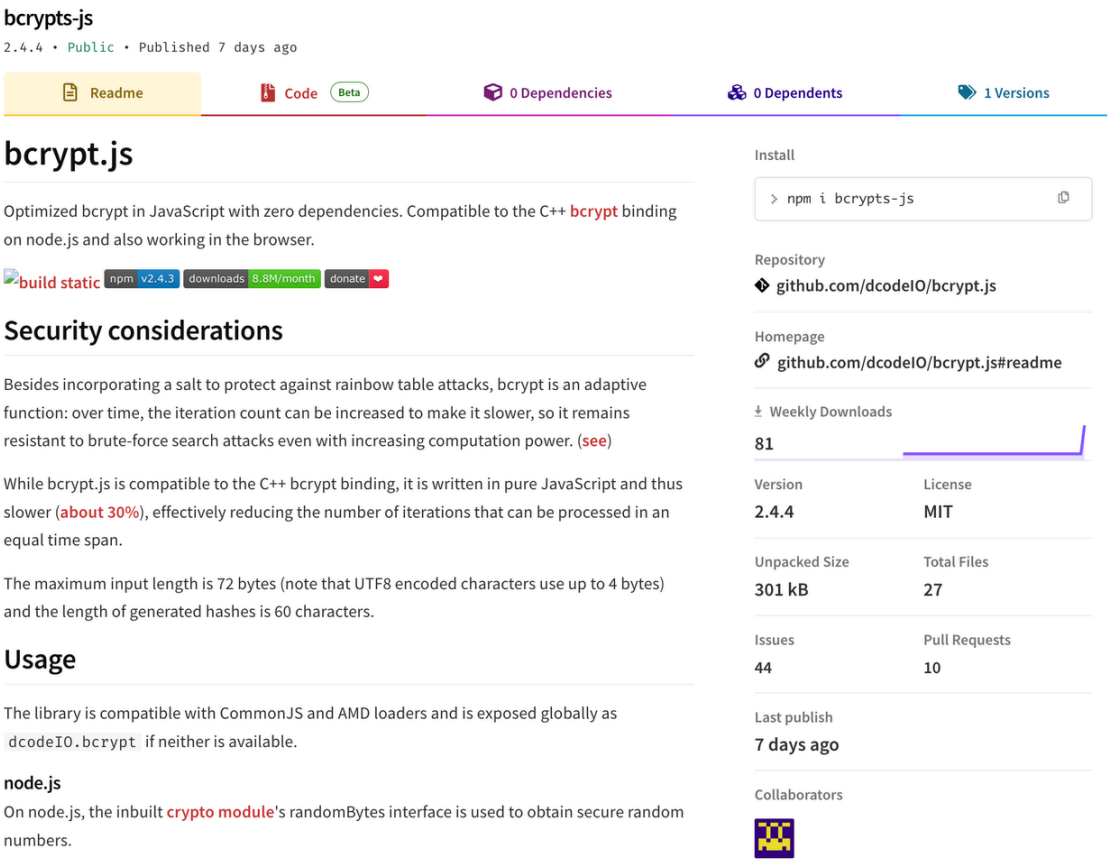
Obfuscated JavaScript found in an otherwise obfuscated passports-js source file (click to enlarge)

After closer investigation, we found that the `passports-js` package was in fact a backdoored copy of `passport`, a legitimate npm package providing a highly popular authentication framework for Express applications. The additional obfuscated line appeared to be the only difference between the two packages. Given this, it would appear that the uploader of `passports-js` was using a namesquatting attack to target would-be `passport` users who misremembered the latter's name.



The npm page for passports-js (click to enlarge)

At the time of this discovery, the uploading user, `superdev727`, had published only one other package to npm: `bcrypts-js`. We determined that `bcrypts-js` was also namesquatting another legitimate npm package, `bcryptjs`, a bcrypt library with 2.1M weekly average downloads at time of writing.



The npm page for bcrypts-js (click to enlarge)

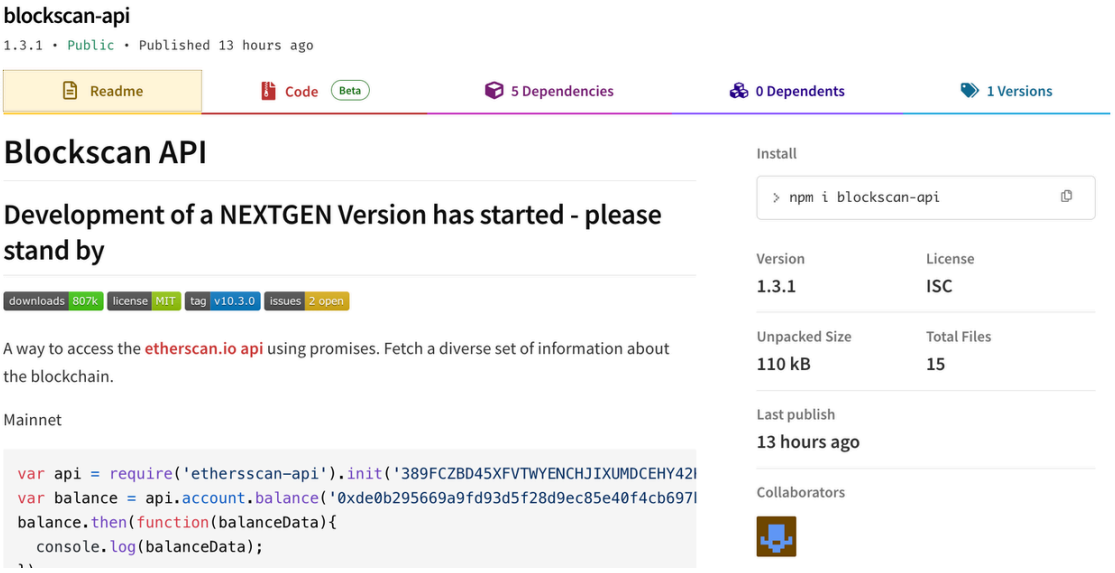
As with `passports-js` and `passport`, the only difference between `bcrypts-js` and `bcryptjs` appeared to be a long, obfuscated line inserted into an unobfuscated source file. We found the obfuscated lines from `passports-js` and `bcrypts-js` to be identical.

As the final entry in this saga, two days later, on September 13, 2024, GuardDog flagged version 1.3.1 of the package `blockscan-api` for review with similar findings. This time, the single obfuscated line was contained in its own source file instead of being wedged in among unobfuscated code.

Obfuscated JavaScript found in a standalone blockscan-api source file (click to enlarge)

We found that `blockscan-api`, like `passports-js` and `bcrypts-js`, is a backdoored copy of another package, `etherscan-api`, which provides an interface to the Etherscan API. The obfuscated line found in `blockscan-api` differed from that in the other two packages. However, we were able to confirm that the two samples are highly similar after deobfuscation, though not without some interesting differences that we describe below.

It is worth noting that `blockscan-api` was published to npm by a different user account, `intelliman`, and also appears to be primarily targeting blockchain developers. At time of discovery, the `intelliman` account had no other published packages.



The npm page for blockscan-api (click to enlarge)

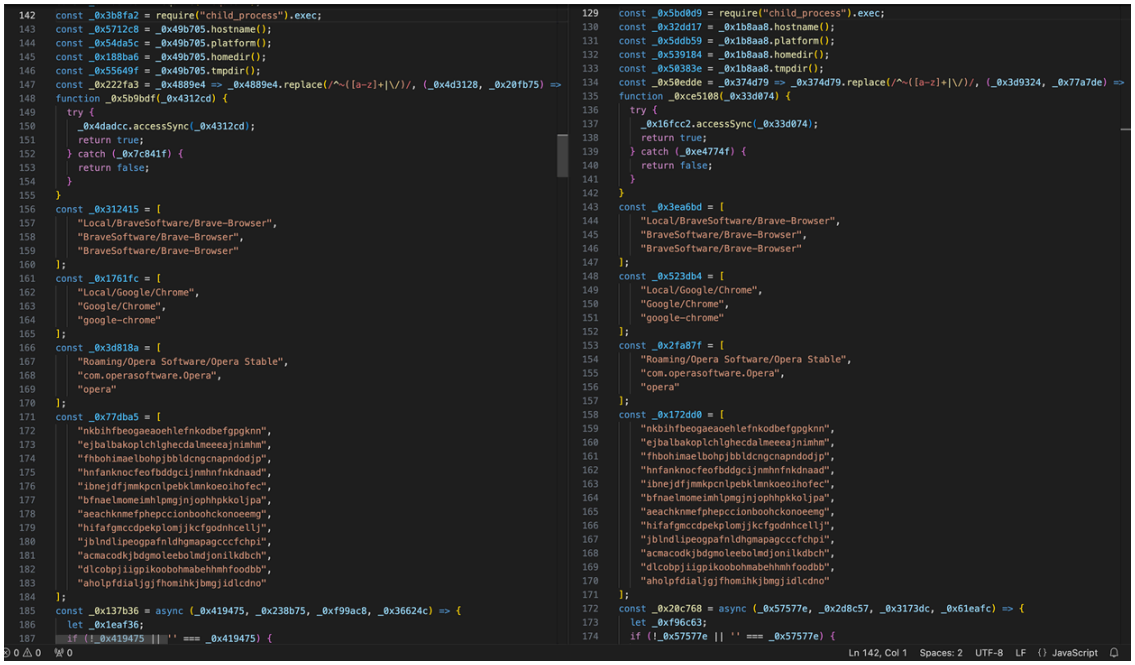
The `passports-js` and `bcrypts-js` packages and the `superdev727` account were removed from npm just after 11pm UTC on September 11, around 12 hours after our initial discovery. Meanwhile, `blockscan-api` and the `intelliman` account remained live for nearly a month, being removed on October 9, 2024 after our report on October 3. GitHub Security Advisories have been released for all three packages. Over their respective lifetimes, `passports-js` was downloaded 118 times, `bcrypts-js` 81 times, and `blockscan-api` at least 124 times, for a total of 323 downloads.

Obfuscated BeaverTail malware

In the npm ecosystem, use of a particular JavaScript obfuscator (available here) is surprisingly common, even among totally benign packages. We found that the two malware samples discovered in `passports-js` / `bcrypts-js` and `blockscan-api` were obfuscated using this common obfuscator. This particular kind of obfuscation can be partially undone easily using free automated tools, allowing us to statically analyze both recovered samples.

What we found was that both obfuscated samples conceal a recent variant of a malware family known as BeaverTail. First identified in late 2023 by researchers at Palo Alto Networks Unit 42, BeaverTail is a JavaScript infostealer and downloader malware prominently used by threat actors connected to the DPRK, in particular as part of the Contagious Interview campaign that targeted developer job applicants.

BeaverTail targets cryptocurrency wallets as well as credit card information stored in browser caches and login keychains on Unix and Windows systems. It then exfiltrates those data to attacker-controlled C2 servers. As described in detail by Unit 42, it also contains logic to download and persistently run a second-stage Python backdoor known as InvisibleFerret from these servers. We observe all characteristic behaviors of BeaverTail in both deobfuscated samples, illustrated via the following images.

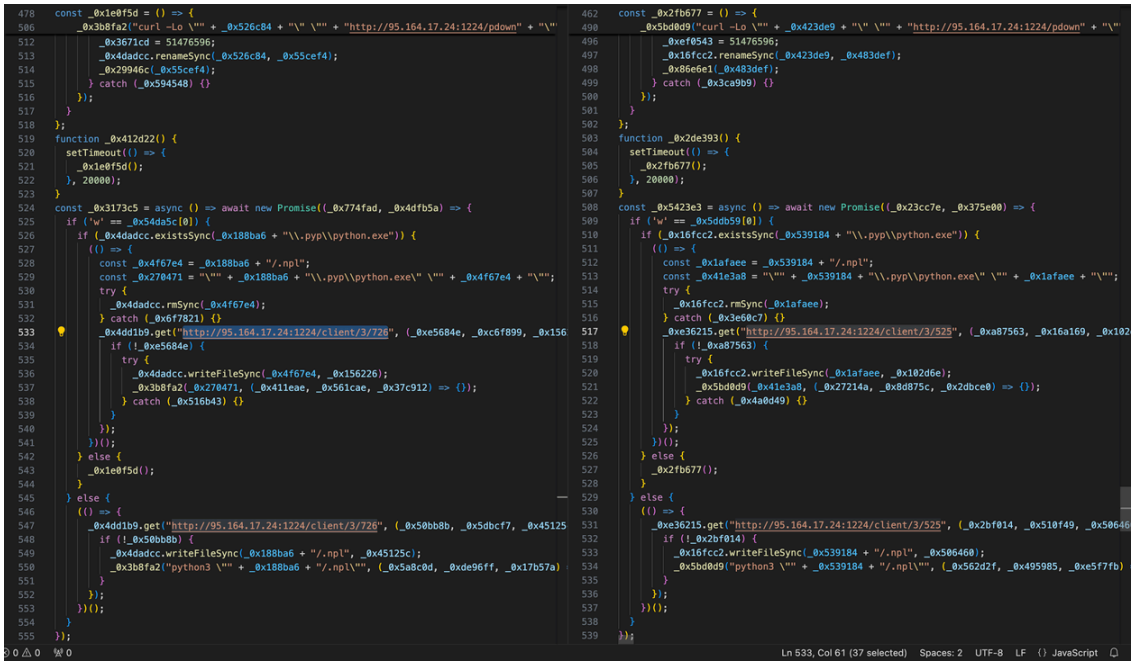


Side-by-side comparison of the passports-js/bcrypts-js and blockscan-api BeaverTail samples showing hardcoded paths for Brave, Google Chrome, and Opera data directories and services as well as IDs for several cryptocurrency wallet browser extensions (click to enlarge)

Side-by-side comparison of the passports-js/bcrypts-js and blockscan-api BeaverTail samples showing exfiltration of data from the Login Keychain (click to enlarge)

Side-by-side comparison of the passports-js/bcrypts-js and blockscan-api BeaverTail samples exfiltration of data from Brave browser caches (click to enlarge)

There are some interesting differences between the two BeaverTail variants. Most notably, they appear to be associated with different threat actor–specified campaign IDs. These IDs are discernable in the URLs shown in the following side-by-side comparison, both of which have the form `http://<C2 server>:1224/client/3/<campaign ID>`. In particular, this is the URL from which BeaverTail sources the first stage of InvisibleFerret to run on the victim's system.



Side-by-side comparison of the passports-js/bcrypts-js and blockscan-api BeaverTail samples showing the presence of different campaign IDs (click to enlarge)

As seen here, the `passports-js` / `bcrypts-js` sample (left) uses campaign ID `726` while the `blockscan-api` sample uses ID `525`. This raises the possibility that different InvisibleFerret variants are being used, with each matched to particular targeted groups.

The campaign ID `525` was recently observed by Stacklok in a new wave of a Contagious Interview–like campaigns targeting blockchain-related developer job applicants. However, it appears that `726` is a previously

unseen campaign ID from this threat actor, indicating the possibility of a new effort to target new segments of Node.js developers.

There may also be a certain amount of refactoring that differentiates the two BeaverTail samples, with two functions in the `passports-js` / `bcrypts-js` sample being slightly more structured than their analogues in the `blockscan-api` sample. These code segments deal with debugging and Firefox data collection, with side-by-side comparisons in the images that follow. It should be noted that these differences may simply be deobfuscation artifacts.

Side-by-side comparison of the passports-js/bcrypts-js and blockscan-api BeaverTail samples showing differences possibly due to refactoring (click to enlarge)

Side-by-side comparison of the passports-js/bcrypts-js and blockscan-api BeaverTail samples showing differences possibly due to refactoring (click to enlarge)

As for the second-stage InvisibleFerret payloads, we were unable to obtain either sample before the C2 infrastructure was taken down.

Links to Contagious Interview

The samples of BeaverTail contained in these packages have several tactics, techniques, and procedures (TTPs) that overlap with those described in public reporting of the Contagious Interview campaign. We have already noted similarities in the distribution method (hosting on npm) and obfuscation of the samples themselves. However, there are some additional observations that allow us to link this activity to Contagious Interview.

Tenacious Pungsan tend to reuse infrastructure for their campaigns. The BeaverTail samples described in this blog all communicate with a web server hosted at the IP address `95.164.17[.]24` on port 1224. In October 2024, this IP was linked to Contagious Interview in a blog that described a new, Qt GUI variant of BeaverTail. Prior to October 2024, two other vendors had linked this IP to DPRK activities.

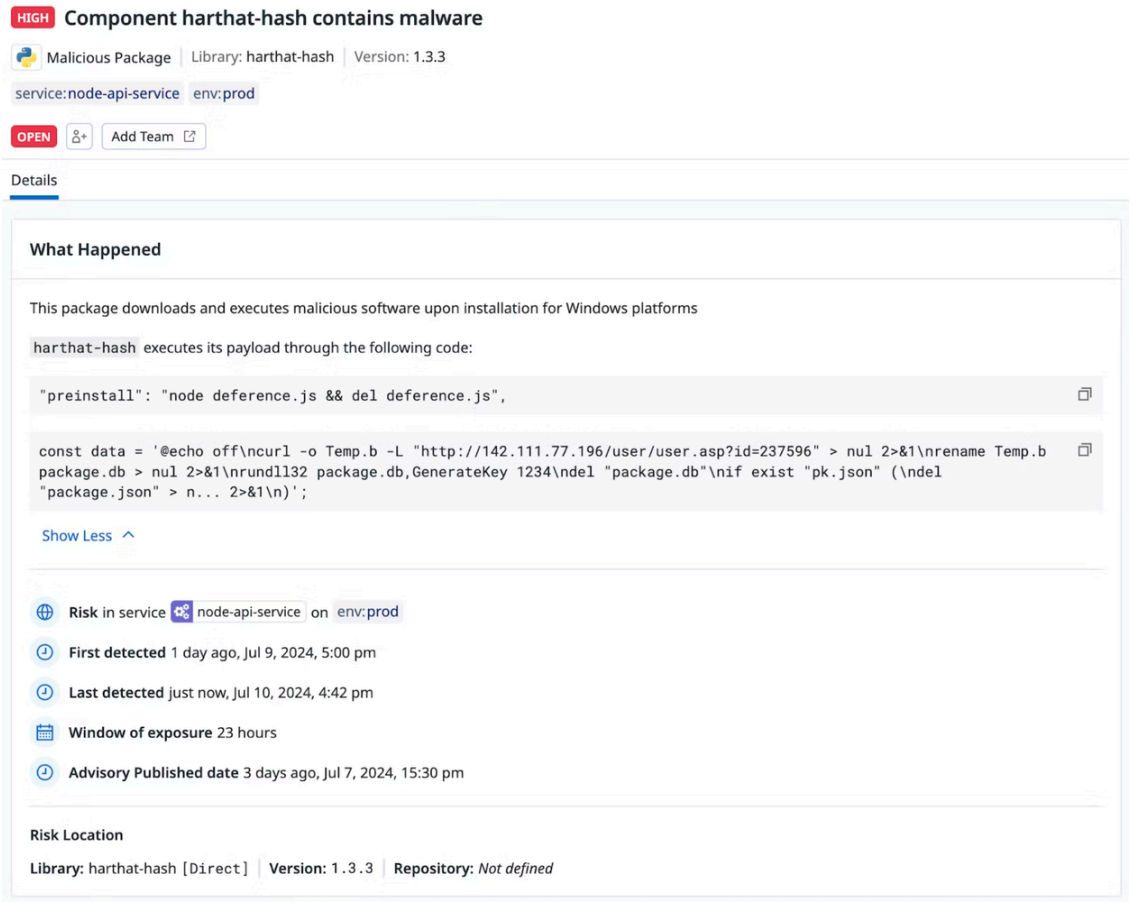
In addition to the IP address, Tenacious Pungsan also reuse the same web directory structure for their C2 server. Exfiltrated files are sent to the URL endpoint `/uploads`, the Python installation is hosted at `/pdown`, and InvisibleFerret is hosted at `/client/<integer>/<3 digit campaign ID>`. This is consistent with the reports linked above.

The infostealer component of BeaverTail targets a specific set of browser extensions associated with cryptocurrency and web3 technologies. This list is consistent across our BeaverTail samples, the Qt GUI variant Unit42 reported on, and the original nodeJS variant also covered by Unit42. Similarly, these samples all attempt to extract the macOS Login Keychain.

The above points allow us to assess with high confidence that these samples are indeed BeaverTail and are being distributed as part of the Contagious Interview campaign.

How Datadog can help

Datadog Software Composition Analysis (SCA) customers can verify whether any of these packages are currently installed in their infrastructure by running this query in the Library Risks explorer: `library_name:(passports-js OR bcrypts-js OR blockscan-api) Status:Open`. If your system is impacted, it is important to take immediate measures such as rotating credentials, isolating the application, and investigating potential spread.



Datadog SCA identifying a malicious dependency (click to enlarge)

In order to enable further research, we have published all affected versions of `passports-js`, `bcrypts-js,` and `blockscan-api` to our public malicious package dataset.

Conclusion

Copying and backdooring legitimate npm packages continues to be a common tactic of threat actors in this ecosystem. These campaigns, along with Contagious Interview more broadly, highlight that individual developers remain valuable targets for these DPRK-linked threat actors.

Indicators of compromise

IP addresses	Purpose	Note
`95.164.17[.]24`	Data exfiltration, InvisibleFerret download	Reused from previous campaign documented by Unit42

NPM authors	Email	Packages published
`superdev727`	`austin27ahn@outlook.com`	`passports-js`, `bcrypts-js`
`intelliman`	`g65492036@gmail.com`	`blockscan-api`
