NODE.JS SECURITY DONE RIGHT

Tips and Tricks They Won't Teach You in School

Liran Tal
R&D Team Lead for a Full-Stack Technology Web Marketplace
Hello!

I am Liran Tal
R&D Team Lead for a Full-Stack Technology Web Marketplace

Hewlett Packard Enterprise
Open Source Evangelist
1. Node.js background
2. Security Horror Stories in Node.js
3. Tips & Recipes
   - Security by HTTP Headers
   - Secure Session Cookies
   - NoSQL Injection
   - Regular Expressions DOS attack
   - Dependencies Vuln’ Scanning
   - Security as a Service
The Big Bang of JavaScript
Node.js Born in 2009

- Open Source
- Cross-Platform
- Asynchronous JavaScript Runtime
Ryan Dahl was inspired to create Node.js after seeing a file upload progress bar on Flickr.

source: https://en.wikipedia.org/wiki/Node.js
By 2011

- Node.JS 0.1.14
- Package Management (npm)
Node.JS Rapid Adoption
Node.JS is JavaScript
JavaScript is Everywhere
2015 GitHub Developer Survey

50,000 World Wide Software Engineers
JavaScript wins Backend and Frontend popularity
# Most Popular Technologies per Dev Type

<table>
<thead>
<tr>
<th>Category</th>
<th>Full-Stack</th>
<th>Front-End</th>
<th>Back-End</th>
<th>Mobile</th>
<th>Math &amp; Data</th>
<th>Students</th>
</tr>
</thead>
<tbody>
<tr>
<td>JavaScript</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>90.5%</td>
</tr>
<tr>
<td>Angular</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>35.8%</td>
</tr>
<tr>
<td>PHP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>35.2%</td>
</tr>
<tr>
<td>Node.js</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31.8%</td>
</tr>
<tr>
<td>SQL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>25.4%</td>
</tr>
<tr>
<td>WordPress</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>24.1%</td>
</tr>
<tr>
<td>Java</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16.8%</td>
</tr>
<tr>
<td>C#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16.0%</td>
</tr>
<tr>
<td>React</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13.4%</td>
</tr>
<tr>
<td>Other</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13.2%</td>
</tr>
</tbody>
</table>
JavaScript wins most open source projects
### III. Top Tech on Stack Overflow

<table>
<thead>
<tr>
<th>Technology</th>
<th>Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>JavaScript</td>
<td>62,588</td>
</tr>
<tr>
<td>Java</td>
<td>55,134</td>
</tr>
<tr>
<td>Android</td>
<td>43,251</td>
</tr>
<tr>
<td>Python</td>
<td>42,918</td>
</tr>
<tr>
<td>C#</td>
<td>41,624</td>
</tr>
<tr>
<td>PHP</td>
<td>32,247</td>
</tr>
<tr>
<td>JQuery</td>
<td>25,241</td>
</tr>
<tr>
<td>C++</td>
<td>24,959</td>
</tr>
<tr>
<td>HTML</td>
<td>24,656</td>
</tr>
<tr>
<td>iOS</td>
<td>23,599</td>
</tr>
<tr>
<td>CSS</td>
<td>19,912</td>
</tr>
<tr>
<td>C</td>
<td>14,022</td>
</tr>
</tbody>
</table>
Security Horror Stories in Node.JS
By January 2015

rimrafall package published to npm
rimrafall

- npm pre-install script

$ rm -rf /*
`rimrafall` has been taken off the @npmjs registry. Thanks to all who let us know about it.
Fishing Attacks, npm Style
validator.js

- helps validate and sanitize strings
String validation and sanitization

- 1,064 commits
- 1 branch
- 100 releases
- 117 contributors
$ npm install validator.js --save
validator.js

!=

validator
malicious modules
of similar names
malicious modules of similar names

3,500,000 socket.io

2,000 socketio
malicious modules of similar names

11,000,000 uglify-js
50,000 uglifyjs
Failing to educate the younger generation
How to Build a WI-FI Dashboard Using Node.js and Ractive.js

Marcello La Rocca  December 10, 2015  +12

This article was peer reviewed by Marc Towler. Thanks to all of SitePoint’s peer reviewers for making SitePoint content the best it can be!
seemingly innocent tutorial to learn from
function getWifiStatus(response, onSuccess, onError) {

    child_process.exec(CONFIG.wifiCommand, function execWifiCommand(err, stdout, var wifi;

    if (err) {
        console.log('child_process failed with error code: ' + err.code);
        onError(response, WIFI_ERROR_MESSAGE);
    } else {
        try {
            wifi = CONFIG.wifiProcessFunction(stdout);
            onSuccess(response, JSON.stringify(wifi));
        } catch (e) {
            console.log(e);
            onError(response, WIFI_ERROR_MESSAGE);
        }
    }
});
function getWiFiStatus(response, onSuccess, onError) {

    child_process.exec(CONFIG.wifiCommand, function execWiFiCommand(err, stdout, stderr) {
        var wifi;

        if (err) {
            console.log('child_process failed with error code: ' + err.code);
            onError(response, WIFI_ERROR_MESSAGE);
        } else {
            try {
                wifi = CONFIG.wifiProcessFunction(stdout);
                onSuccess(response, JSON.stringify(wifi));
            } catch (e) {
                console.log(e);
                onError(response, WIFI_ERROR_MESSAGE);
            }
        }
    });
}
Security by HTTP Headers
Browsers enforce secure connections to the server (HTTPS)
X-FRAME-OPTIONS

Clickjacking protection by not rendering content in iframes
CONTENT-SECURITY-POLICY

Whitelist trusted content, and services
X-XSS-PROTECTION enables browser XSS filtering

* IE8 | IE9
X-COMENT-TYPE-OPTIONS

*browsers do not sniff MIME responses

*IE8 | Chrome | Safari
Helmet
Securing ExpressJS
Putting it all together with Helmet and ExpressJS

```javascript
var app = express();

app.use(helmet.hsts({
  maxAge: reqDuration,
  includeSubDomains: true
}));

app.use(helmet.frameguard({
  action: 'sameorigin'
}));

app.use(helmet.csp({
  directives: {
    defaultSrc: ['self', 'https://ajax.googleapis.com'],
    scriptSrc: ['self'],
    styleSrc: ['self'],
    childSrc: ['none'],
    objectSrc: ['none'],
    formSrc: ['none']
  }
}));

app.use(helmet.xssFilter());

app.use(helmet.noSniff());
```
Putting it all together with Lusca and ExpressJS

```javascript
'use strict';

// Copying example from Lusca's GitHub page
var express = require('express'),
    app = express(),
    session = require('express-session'),
    lusca = require('lusca');

app.use(lusca.csrf());
app.use(lusca.csp({ /* ... */ }));
app.use(lusca.xframe('SAMEORIGIN'));
app.use(lusca.p3p('ABCDEF'));
app.use(lusca.hsts({ maxAge: 31536000 }));
app.use(lusca.xssProtection(true));
app.use(lusca.nosniff());
```
Securing the Cookies
SECURE

cookies sent over HTTPS connections only
httpOnly

cookies are not accessible from JavaScript
var session = require('express-session');

app.use(session({
  cookie: {
    secure: true,
    httpOnly: true
  }
}));
Fingerprinting Node.js
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 4119
ETag: "658045625"
Set-Cookie: connect.sid=s%3AHQrjo6cepxwRJn28dnfno3md.ednWDA NGxMTIP%2Fv1
Date: Thu, 27 Feb 2014 14:30:43 GMT
Connection: keep-alive
var session = require('express-session');

app.use(session({
    name: 'CristianoRonaldo7';
}));
Fun with Headers
X-hacker: If you're reading this, you should visit automattic.com/jobs and apply to join the fun, mention this header.

X-Pingback: http://wordpress.com/xmlrpc.php

Link: <http://wp.me/1>; rel=shortlink

X-nananana: Batoche

Content-Encoding: gzip
Content-Type: text/html; charset=UTF-8
Vary: Accept-Encoding
Set-Cookie:
Content-Encoding: gzip
Server: ‘; DROP TABLE servertypes; —
Content-Length: 18033
Date: Wed, 15 Aug 2012 13:30:32 GMT
Connection: keep-alive
3 noSQL Injections
Creating TRUE SQL statements

```sql
SELECT * FROM users WHERE username='' OR 1=1 -- AND password=''```

Creating TRUE SQL statements

```javascript
User.find({username: userUsername, password: userPass});
```
show me the code...

Live Demo!
No HTTP body in ExpressJS
it relies on bodyParser lib
ExpressJS uses bodyParser library to access HTTP body payload.
ExpressJS uses the `body-parser` library to access the HTTP body payload.
Creating TRUE SQL statements

curl -X POST \
   -H "Content-Type: application/json" \
   --data '{"username":{"$gt": ""}, "password":{"$gt": ""}}' \
http://localhost:31337/login
Creating TRUE SQL statements

```bash
15  curl -X POST \\
16   -H "Content-Type: application/json" \\
17   --data '{"username":"$gt": ""}, "password":{"$gt": ""}'} \\
18   http://localhost:31337/login
```

```
lirantal:/workspace/code/injections-nosql (master) $ curl -X POST \
>   -H "Content-Type: application/json" \
>   --data '{"username":"$gt": ""}, "password":{"$gt": ""}'} \
>   http://localhost:31337/login
[[{"_id":"57a85181198d09dc661594ba", "password":"demo", "username":"demo"}]]
```
Validate Input

- Validate Length and Type
- Validate & Sanitize input to expected type
- Parameters Binding
- Security in Depth
ExpressJS uses `bodyParser` library to access HTTP body payload.
ReDoS
Regular Expressions DoS
Requirement:

- Validate the input has at least one ‘a’ character and allow unlimited occurrences
var regex = /^(a+)/$;
3 Months Later...
More work on the feature:

- Different Engineer gets the job
- Requirement changes: Validate the input has **exactly 10 characters of ‘a’**
show me the code...

Live Demo!
//var regex = /(^a+)$/;
var regex = /(^a+){10}$/;
Attacker sends

Array(100).join('a') + '!'
ExpressJS uses neogitator

- parsing the Accept-Language header
- Parameters Binding
app.get('/home', function (req, res) {
  req.acceptsLanguages('en');
  res.status(200).send(obj);
});
10,000,000

negotiator
# Regular Expression Denial of Service

<table>
<thead>
<tr>
<th>Affected package</th>
<th>Vulnerable versions</th>
<th>Latest version</th>
</tr>
</thead>
<tbody>
<tr>
<td>negotiator</td>
<td>&lt;= 0.6.0</td>
<td>0.6.1</td>
</tr>
</tbody>
</table>
Best Practices

diamond Validator.js node.js module

Watch 146  Star 5,983  Fork 562

npm v5.5.0  build passing  coverage 99%  downloads 1M/month
<table>
<thead>
<tr>
<th>Filename</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>isBase64.js</td>
<td>Rebase on 4.9.0</td>
</tr>
<tr>
<td>isBefore.js</td>
<td>Keep things simple</td>
</tr>
<tr>
<td>isBoolean.js</td>
<td>Keep things simple</td>
</tr>
<tr>
<td>isByteLength.js</td>
<td>Update generated code</td>
</tr>
<tr>
<td>isCreditCard.js</td>
<td>Additional stuff for #539</td>
</tr>
<tr>
<td>isCurrency.js</td>
<td>Prefer template strings</td>
</tr>
<tr>
<td>isDataURL.js</td>
<td>Update dependencies and apply lint fixes</td>
</tr>
<tr>
<td>isDate.js</td>
<td>Update generated code</td>
</tr>
<tr>
<td>isDecimal.js</td>
<td>Keep things simple</td>
</tr>
<tr>
<td>isDivisibleBy.js</td>
<td>Keep things simple</td>
</tr>
<tr>
<td>isEmail.js</td>
<td>Additional stuff for #532</td>
</tr>
<tr>
<td>isFQDN.js</td>
<td>Allow &gt;1 underscore in hostnames, closes #510</td>
</tr>
</tbody>
</table>
Best Practices

- safe-regexp node.js module
- checks regex complexity/backtracking vulnerability
var regex = /^\s*(\S+)?:\?(:-(\S+))?\s*(:\?(.*))?$/;
var regexSafe = safeRegex(regex);
console.log(regexSafe);
Best Practices

◇ OWASP Validation RegEx Repo
5 Vulnerability Scan
ask yourself

Are my dependencies vulnerable?
snyk

- check cve db for known issues
- check installed node_modules dir
- provides patch-level fix
- provides interactive patch wizard
"version": "1.0.0",
"description": ",",
"main": "index.js",
"scripts": {
  "test": "echo "Error: no test specified" && exit 1"
},
"keywords": [],
"author": ",",
"license": "ISC",
"dependencies": {
  "mocha": "^2.5.0",
  "nsp": "^2.6.1",
  "swig": "^1.4.0"
}
lirantal:~/workspace/code/snyk (master) $ snyk test
X High severity vulnerability found on minimatch@0.3.0
  - desc: Regular Expression Denial of Service
  - from: snyk@1.0.0 > mocha@2.5.0 > glob@3.2.11 > minimatch@0.3.0
Upgrade direct dependency mocha@2.5.0 to mocha@3.0.0 (triggers upgrades to glob@7).

X Low severity vulnerability found on uglify-js@2.4.24
  - desc: Regular Expression Denial of Service
  - from: snyk@1.0.0 > swig@1.4.0 > uglify-js@2.4.24
No direct dependency upgrade can address this issue.
Run `snyk wizard` to explore remediation options.

Tested 64 dependencies for known vulnerabilities, found 2 vulnerabilities, 2 vuln

Run `snyk wizard` to address these issues.
nsp

- check cve db for known issues
- check installed node_modules dir
```
<table>
<thead>
<tr>
<th>Name</th>
<th>minimatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installed</td>
<td>0.3.0</td>
</tr>
<tr>
<td>Vulnerable</td>
<td>&lt;=3.0.1</td>
</tr>
<tr>
<td>Patched</td>
<td>&gt;=3.0.2</td>
</tr>
<tr>
<td>Path</td>
<td>snyk@1.0.0 &gt; mocha@2.5.3 &gt; glob@3.2.11 &gt; minimatch@0.3.0</td>
</tr>
</tbody>
</table>
```
```
<table>
<thead>
<tr>
<th>Name</th>
<th>uglify-js</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installed</td>
<td>2.4.24</td>
</tr>
<tr>
<td>Vulnerable</td>
<td>&lt;=2.6.0</td>
</tr>
</tbody>
</table>
```
shrinkwrap

- pin-down dependencies
- pin-down devDependencies
- ship with tested packages
- avoid surprises in production build
{
  "name": "snyk",
  "version": "1.0.0",
  "dependencies": {
    "mocha": {
      "version": "2.5.0",
      "from": "mocha@2.5.0",
      "resolved": "https://registry.npmjs.org/mocha/-/mocha-2.5.0.tgz",
      "dependencies": {
        "commander": {
          "version": "2.3.0",
          "from": "commander@2.3.0",
          "resolved": "https://registry.npmjs.org/commander/-/commander-2.3.0.tgz"
        },
        "debug": {
          "version": "2.2.0",
          "from": "debug@2.2.0",
          "resolved": "https://registry.npmjs.org/debug/-/debug-2.2.0.tgz",
          "dependencies": {
            "ms": {
              "version": "0.7.1",
              "from": "ms@0.7.1",
              "resolved": "https://registry.npmjs.org/ms/-/ms-0.7.1.tgz"
            }
          }
        }
      }
    }
  }
}
SecurityOps

Integrated Security into your build pipeline
Security as a Service
david-dm

- monitor nodejs dependencies
- check installed node_modules dir
BOWER - BOWER 1.7.9

The browser package manager

<table>
<thead>
<tr>
<th>DEPENDENCIES</th>
<th>DEVDEPENDENCIES</th>
</tr>
</thead>
</table>

49 Dependencies total  30 Up to date  8 Pinned, out of date  9 Out of date

SECURITY VULNERABILITIES IN DEPENDENCIES

Advisories from the Node Security Project

<table>
<thead>
<tr>
<th>DEPENDENCY</th>
<th>REQUIRED</th>
<th>STABLE</th>
<th>LATEST</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbrev</td>
<td>^1.0.5</td>
<td>1.0.9</td>
<td>1.0.9</td>
<td></td>
</tr>
<tr>
<td>archy</td>
<td>1.0.0</td>
<td>1.0.0</td>
<td>1.0.0</td>
<td></td>
</tr>
</tbody>
</table>
Bithound.io

- monitor nodejs dependencies
- lint / static code analysis
## Dependencies Analysis

<table>
<thead>
<tr>
<th>Category</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Packages</td>
<td>111</td>
</tr>
<tr>
<td>Priority</td>
<td>47</td>
</tr>
<tr>
<td>Insecure</td>
<td>11</td>
</tr>
<tr>
<td>Disallowed</td>
<td>0</td>
</tr>
<tr>
<td>Deprecated</td>
<td>2</td>
</tr>
<tr>
<td>Outdated</td>
<td>47</td>
</tr>
</tbody>
</table>

### Priority Dependencies

**mocha**

- **status**: INSECURE
- **version**: Required: 2.5.0 | Stable: 3.0.2
- **License**: MIT

### Code Analysis

<table>
<thead>
<tr>
<th>Category</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analyzed Files</td>
<td>134</td>
</tr>
<tr>
<td>Priority Files</td>
<td>0</td>
</tr>
<tr>
<td>Test Files</td>
<td>31</td>
</tr>
<tr>
<td>Blacklisted Files</td>
<td>0</td>
</tr>
</tbody>
</table>

### Files

- **modules/users/tests/server/user.server.model.tests.js**  
  Last updated Apr 29 2016

**Analysis**

- **Duplicate Functions**: 12
- **Errors Ignored**: 2
Summary:

1. Helmet or Lusca for secure HTTP headers
2. Obfuscate the session name
3. Validate and Sanitize req.body params to NoSQL
Summary:

4. Use validator.js for regex
5. Dependencies check with snyk, and nsp
6. SaaS Security with bithound.io and david-dm
Thanks!

Any questions?

- liran.tal@hpe.com
- GitHub