Dependencies With Known Vulnerabilities

In this lesson, we'll study a few open-source libraries that have dependencies with known vulnerabilities.

Chances are that the application you're working on *right now* depends on a plethora of open-source libraries: ExpressJS, a popular web framework for NodeJS, depends on 30 external libraries, and those libraries depend on external libraries, and those...we could go on forever. As a simple exercise, I tried to install a brand-new version of ExpressJS in my system, with interesting results:

```
$ npm install express
+ express@4.17.1
added 50 packages from 37 contributors and audited 127 packages in 9.072s
found 0 vulnerabilities
```

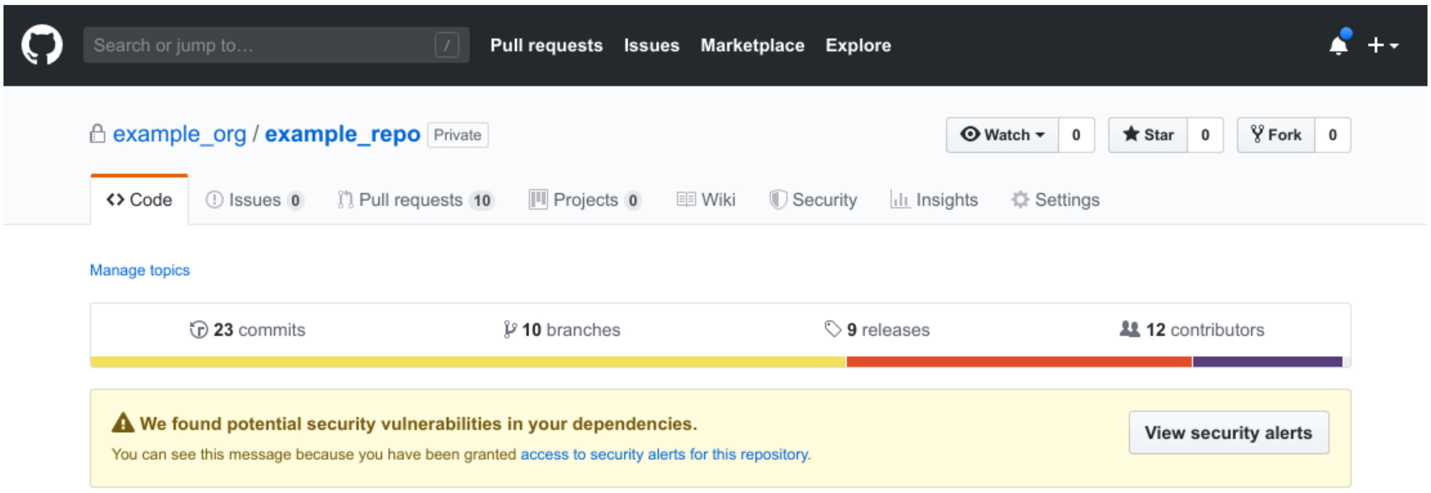
Just by installing the latest version of ExpressJS, I've included 50 libraries in my codebase. Is that inherently bad? Not at all, but it presents a security risk; the more code we write or use, the larger the attack surface for malicious users becomes.

One of the biggest risks when using a plethora of external libraries is not following up on updates when they are released. It isn't so bad to use open-source libraries, after all, they're probably safer than most of the code we write ourselves. But forgetting to update them, especially when a security fix is released, is a genuine problem we face every day.

Luckily, programs like npm provide tools to identify outdated packages with known vulnerabilities. We can simply try to install a dependency with a known vulnerability and run `npm audit fix`, and npm will do the rest for us.

```
$ npm install lodash@4.17.11
+ lodash@4.17.11
added 1 package from 2 contributors and audited 288 packages in 1.793s
found 1 high severity vulnerability
  run `npm audit fix` to fix them, or `npm audit` for details
$ npm audit


                       === npm audit security report ==
=



# Run  npm update lodash --depth 1  to resolve 1 vulnerability
┌───────────────┬──────────────────────────────────────────────────────────────
┐
│ High          │ Prototype Pollution
                            │
├───────────────┼──────────────────────────────────────────────────────────────
┤
│ Package       │ lodash
                            │
├───────────────┼──────────────────────────────────────────────────────────────
┤
│ Dependency of │ lodash
                            │
├───────────────┼──────────────────────────────────────────────────────────────
┤
│ Path          │ lodash
                            │
├───────────────┼──────────────────────────────────────────────────────────────
┤
│ More info     │ https://npmjs.com/advisories/1065
          │
└───────────────┴──────────────────────────────────────────────────────────────
┘



found 1 high severity vulnerability in 1 scanned package
  run `npm audit fix` to fix 1 of them.
$ npm audit fix
+ lodash@4.17.15
updated 1 package in 0.421s
fixed 1 of 1 vulnerability in 1 scanned package
```

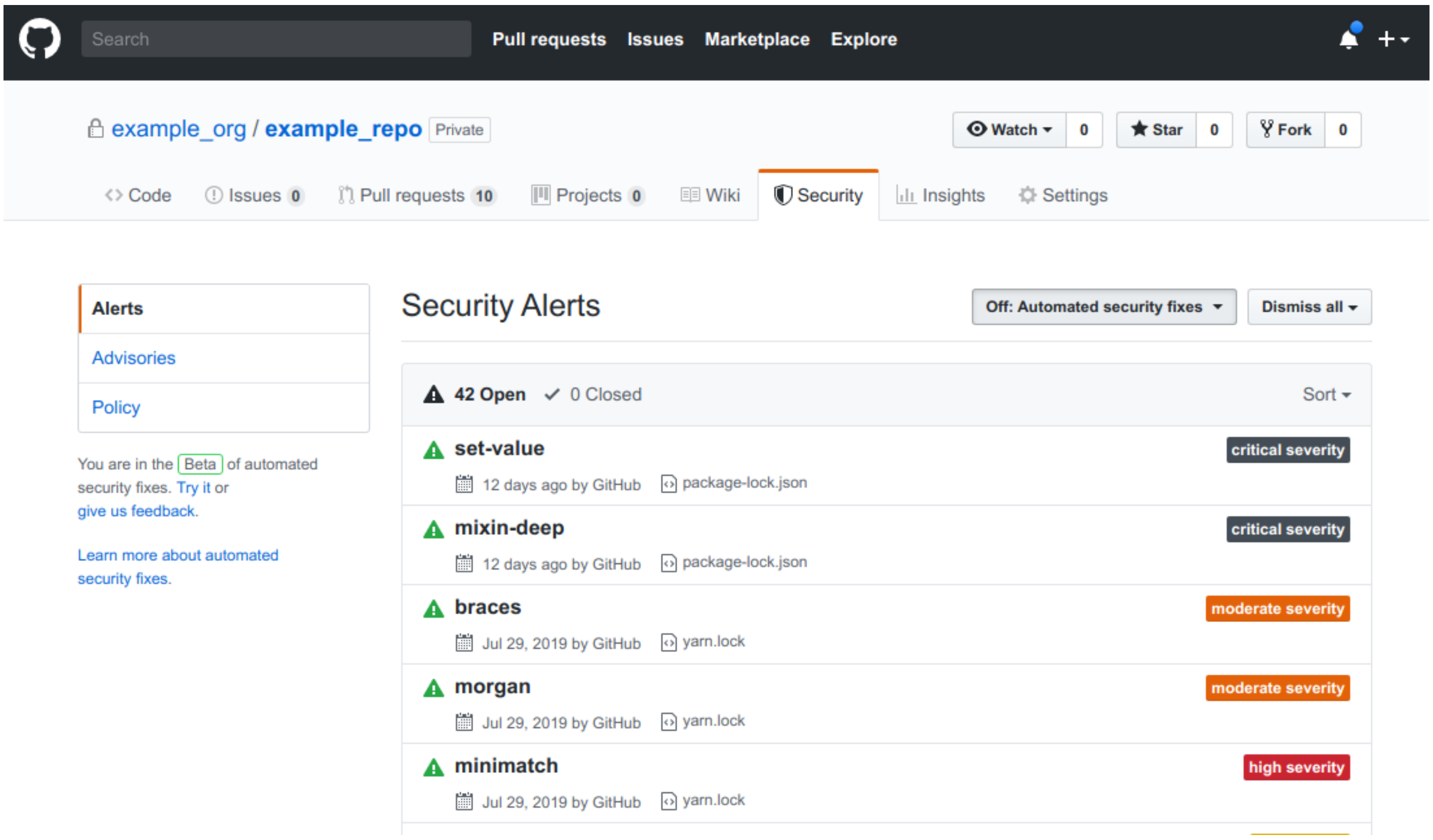
If you’re not using JavaScript and npm, you can always rely on external services to scan your software and point out any library with known vulnerabilities. GitHub offers this service for all their repositories, and you might find it convenient if your codebase is already hosted there.



We found potential security vulnerabilities in your dependencies.
You can see this message because you have been granted access to security alerts for this repository.

The alert banner prominently displayed in a vulnerable repository

GitHub will also send you an email every time a dependency with a known vulnerability is detected, so you can head over to the repository and look at the problem in detail.



You are in the Beta of automated security fixes. Try it or give us feedback.

Learn more about automated security fixes.

Security Alerts

42 Open ✓ 0 Closed

set-value — critical severity
12 days ago by GitHub — package-lock.json

mixin-deep — critical severity
12 days ago by GitHub — package-lock.json

braces — moderate severity
Jul 29, 2019 by GitHub — yarn.lock

morgan — moderate severity
Jul 29, 2019 by GitHub — yarn.lock

minimatch — high severity
Jul 29, 2019 by GitHub — yarn.lock

The vulnerability details presented by GitHub

If you prefer using a different platform, you could try gitlab.com. In 2018 it acquired Gemnasium, a product that offered vulnerability scanning, in order to compete with GitHub's offering. If you prefer to use a tool that does not require code hosting, snyk.io would probably be your best bet. It's trusted by massive companies like Google, Microsoft and SalesForce, and offers different tools for your applications, not just dependency scanning.

In the next lesson, we’ll look at an interesting service that allows you to see if your username/password was part of (a) security breach(es).