

# Checando suas dependências

Você confia nas dependências das suas dependências?

# Inajara Leppa

**Consultora de Desenvolvimento na  
ThoughtWorks e representante do pilar de  
Justiça Econômica e Social em Porto Alegre**

**ThoughtWorks®**

©ThoughtWorks 2018 Commercial in Confidence

ThoughtWorks



# Lucas Vieira

Consultor de Desenvolvimento na ThoughtWorks

ThoughtWorks®

©ThoughtWorks 2018 Commercial in Confidence



# OWASP TOP 10

A9:2017: Utilizando componentes com vulnerabilidades conhecidas

## Sua aplicação está vulnerável?

- Se você não souber as versões de todos os componentes usados. Isso inclui componentes que você usa diretamente, bem como dependências aninhadas.
- Se você não verificar vulnerabilidades regularmente e se inscrever em boletins de segurança relacionados aos componentes que você usa.
- Se você não corrigir ou atualizar a plataforma, as estruturas e as dependências subjacentes de maneira oportuna e baseada em risco.
- Se os desenvolvedores de software não testarem a compatibilidade de bibliotecas atualizadas, atualizadas ou corrigidas.

*"Security is always excessive  
until it is not enough."*

**– Robbie Sinclair**

**Head of Security, Country Energy, NSW Australia**



# npm audit

# O que é?

- Comando que analisa as dependências da sua aplicação, buscando vulnerabilidades conhecidas pela comunidade.
- npm audit checa dependencias diretas, devDependencies, bundledDependencies, e optionalDependencies, mas não checa peerDependencies.
- Está disponível a partir da versão 6 do npm

# Como faz?

```
> npm audit [--json|--parseable]
```

`--json` Relatório detalhado em formato JSON

`--parseable` Relatório detalhado em formato de texto



Manual Review Some vulnerabilities require your attention to resolve Visit <a href="https://go.npm.me/audit-guide">https://go.npm.me/audit-guide</a> for additional guidance	
<b>Moderate</b>	<b>Prototype pollution</b>
Package	hoek
Patched in	> 4.2.0 < 5.0.0    >= 5.0.3
Dependency of	numbat-emitter
Path	numbat-emitter > request > hawk > boom > hoek
More info	<a href="https://nodesecurity.io/advisories/566">https://nodesecurity.io/advisories/566</a>

### Vulnerabilidade que necessita de revisão manual

<pre> === npm audit security report ===  # Run npm install chokidar@2.0.3 to resolve 1 vulnerability SEMVER WARNING: Recommended action is a potentially breaking change </pre>	
<b>Low</b>	<b>Prototype Pollution</b>
Package	deep-extend
Dependency of	chokidar
Path	chokidar > fsevents > node-pre-gyp > rc > deep-extend
More info	<a href="https://nodesecurity.io/advisories/612">https://nodesecurity.io/advisories/612</a>

### Vulnerabilidade com ação recomendada para resolução

severity low

# Prototype Pollution

deep-extend

Advisory

Versions

## Overview

Versions of `deep-extend` before 0.5.1 are vulnerable to prototype pollution.

## Remediation

Update to version 0.5.1 or later.

## Resources

- [HackerOne Report](#)

## Advisory timeline



### Published

Advisory published  
Apr 25th, 2018



### Reported

Initial report by  
Olivier Arteau  
(HoLyVieR)  
Apr 24th, 2018

Severidade	Ação Recomendada
Crítica (Critical)	Endereçar imediatamente
Alta (High)	Endereçar o quanto antes
Moderada (Moderate)	Endereçar quando possível
Baixa (Low)	Endereçar sem pressa

É possível passar o nível mínimo de severidade no npm config (audit-level)

# npm audit fix

# O que é?

- Comando que instala automaticamente as atualizações para as dependências vulneráveis que não necessitam de revisão manual.

# Como faz?

```
> npm audit fix [--only|--dry-run]
```

`--only=prod` executa apenas nas dependências diretas

`--dry-run` mostra prévia do que vai acontecer ao executar o comando



# #ficadica



- Se o projeto utiliza um registry privado para o npm (ex: jfrog artifactory), é necessário passar o registry público junto do comando e o audit será executado apenas nos repositórios públicos do npm.

ex: `npm audit --registry=https://registry.npmjs.org`

- Não recomendamos utilizar o comando de fix caso sua aplicação utilize um registry privado, pois isso irá alterar a url da dependência para apontar para o registry público. A solução para isso é executar os comandos recomendados pelo audit manualmente.

- Diferente do comando de fix, o comando de audit ainda não suporta verificar apenas dependências de produção e essa verificação deve ser feita manualmente.
- Como forma de trazer esse olhar de segurança para o time, adicionamos essa verificação em nossa pipeline, que é executada 2x por dia e caso encontre vulnerabilidades envia uma mensagem em nosso canal de alertas do time.

# Extras

- <https://github.com/thoughtworks/talisman>: A tool to detect and prevent secrets from getting checked in.
- <https://github.com/AGWA/git-crypt>: git-crypt enables transparent encryption and decryption of files in a git repository.
- <https://github.com/zaproxy/zaproxy>: It can help you automatically find security vulnerabilities in your web applications while you are developing and testing your applications. It's also a great tool for experienced pentesters to use for manual security testing.
- <https://yarnpkg.com/lang/en/docs/cli/audit/>: Yarn also has an audit tool, but we didn't test it as much as the npm one.

# Obrigada

**Inajara Leppa** ileppa@thoughtworks.com  
**Lucas Vieira** lucas.f@thoughtworks.com