

SMARTPHONE SÉCURISER

NDONG MBA VALENTINO

CONTENU

- 01** Introduction
- 02** outils et définitions
- 03** types de vulnérabilités
- 04** détection
- 05** solutions
- 06** Conclusion

INTRODUCTION



En cybersécurité, le modèle serveur-client décrit une architecture dans laquelle deux entités interagissent pour échanger des données ou exécuter des opérations sur un réseau



Le modèle client-serveur est donc au cœur des préoccupations en cybersécurité, car une faille sur l'une des deux entités ou dans leur communication peut exposer des données sensibles ou compromettre l'ensemble du système.



outils et définitions

snyk

Snyk analyse automatiquement les projets pour repérer des problèmes comme des librairies obsolètes, des failles connues ou des configurations faibles, tout en proposant des correctifs adaptés. Elle est particulièrement utile pour renforcer la sécurité dans des environnements DevSecOps, en rendant la gestion des risques rapide et accessible pour les équipes techniques.

semgrep

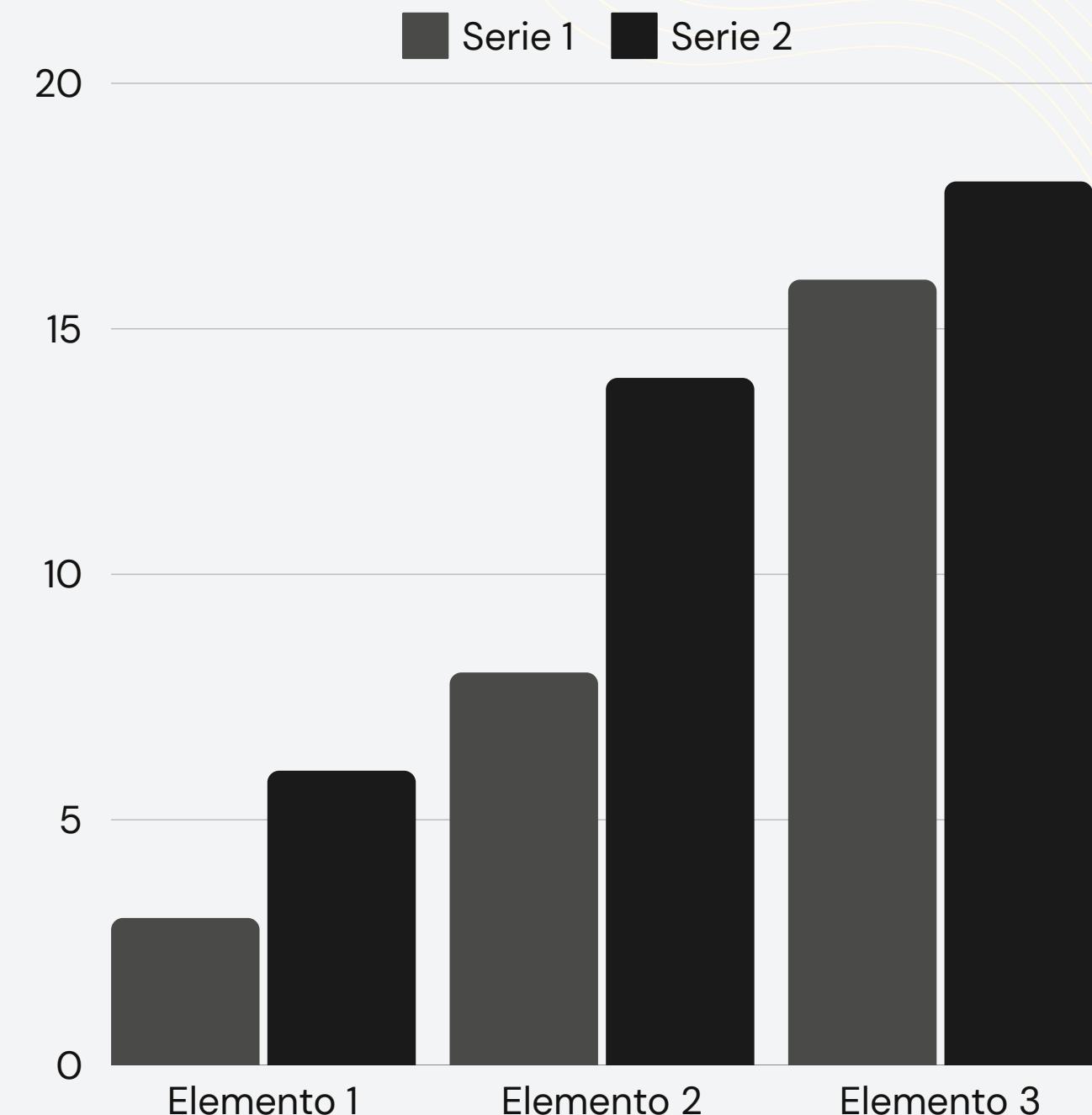
Semgrep est un outil de sécurité statique qui analyse le code source pour détecter des vulnérabilités, des mauvaises pratiques et des problèmes de conformité. Il est rapide, facile à configurer et permet d'écrire des règles personnalisées en fonction des besoins du projet.

retire.js

Retire.js est un outil open-source qui détecte les vulnérabilités dans les dépendances JavaScript côté client et serveur. Il analyse les fichiers et les dépendances (comme celles de npm, Bower, ou des scripts dans le navigateur) pour identifier les versions obsolètes ou contenant des failles connues. L'outil est léger, rapide et peut être intégré dans les pipelines CI/CD pour améliorer la sécurité des projets JavaScript en continu.

TYPES DE VULNERABILITÉS

La classification des vulnérabilités en cybersécurité permet d'évaluer et de prioriser les failles selon leur gravité et leur impact, afin de mieux gérer les risques. Les vulnérabilités sont classées par niveau de gravité (critique, élevée, moyenne, basse) et évaluées avec des systèmes comme le CVSS (score de 0 à 10). Chaque vulnérabilité se voit attribuer un identifiant unique via le CVE pour faciliter son suivi. La classification prend également en compte le contexte d'exploitation (accès nécessaire, type de dommage) et l'impact potentiel (fuite de données, exécution de code, élévation de privilèges). Cette méthode permet de répondre rapidement aux failles critiques tout en optimisant les ressources de remédiation.



Detection

Snyk | SECURITY

Snyk Vulnerability Database / Maven / org.webjars.npm:underscore

Arbitrary Code Injection

Affecting [org.webjars.npm:underscore](#) package, versions [0,1.13.1]

INTRODUCED: 2 MAR 2021 [CVE-2021-23358](#) [CWE-94](#)

How to fix?
Upgrade `org.webjars.npm:underscore` to version 1.13.1 or higher.

Severity: 5.5 (MEDIUM)

CVSS assessment made by Snyk's Security Team. [Learn more](#)

Threat Intelligence

Exploit Maturity: PROOF OF CONCEPT
EPSS: 0.69% (81st percentile)

Overview

`org.webjars.npm:underscore` is a JavaScript's functional programming helper library.
Affected versions of this package are vulnerable to Arbitrary Code Injection via the `template` function, particularly when the `variable` option is taken from `_.templateSettings.variable` as it is not sanitized.

PoC

```
const _ = require('underscore');
_.templateSettings.variable = "a = this.process.mainModule.require('child_process').execSync('touch HELLO')";
_.template(...);
```

ORGANISATION

v valentinowyhnel

Tableau de bord

Projets

Intégrations

Membres

Paramètres

Mises à jour des produits

Aide

NDONG MBA Vale...

Search by package name or CVE

Analysé de code

Aperçu Histoire Paramètres

GRAVITÉ

Haut: 0
Moyen: 2
Faible: 0

SCORE DE PRIORITÉ

Noté entre 0 et 1000: 600

STATUT

Ouvrir: 2
Ignoré: 0

LANGUES

Exposition d'informations (CODE SNYK | CWE-200)

SCORE 600

```
3 const session = require('express-session');
4 const fs = require('fs');
5
6 // Initialize the app
7 const app = express();
```

Désactivez l'en-tête X-Powered-By pour votre [application Express](#) (envisagez d'utiliser le middleware Helmet), car il expose des informations sur le framework utilisé aux attaquants potentiels.

application.js

1 étape dans 1 fichier

Ouvert sur GitHub

ORGANISATION

v valentinowyhnel

Tableau de bord

Projets

Intégrations

Membres

Paramètres

Mises à jour des produits

Aide

valentinowyhnel > Projets > valentinowyhnel/vuln_analyse principal

Analysé de code

Aperçu Histoire Paramètres

Falsification de requête intersite (CSRF) (CODE SNYK | CWE-352)

SCORE 600

```
3 const session = require('express-session');
4 const fs = require('fs');
5
6 // Initialize the app
7 const app = express();
```

La protection CSRF est désactivée pour votre [application Express](#). Cela permet aux attaquants d'exécuter des requêtes au nom d'un utilisateur.

application.js

1 étape dans 1 fichier

Découvrez ce type de vulnérabilité et comment y remédier

dad [En fonction] - Oracle VirtualBox

Fichier Machine Écran Entrée Périphériques Aide

Kali Linux ZAP - Download GitHub - zaproxy Kali Linux (venv)root@dad: /home/dad

Fichier Actions Éditer Vue Aide

root@dad: /home/dad x root@dad: /home/dad/web-app-vulnerability-analysis--project- x (venv)root@dad: /home/dad x https://secu

```
root@dad:~/home/dad/web-app-vulnerability-analysis--project-/app.js
  > javascript.express.security.audit.express-check-csurf-middleware-usage.express-check-csurf-middleware-usage
    A CSRF middleware was not detected in your express application. Ensure you are either using one such
    as `csurf` or `csrf` (see rule references) and/or you are properly doing CSRF validation in your
    routes with a token or cookies.
    Details: https://sg.run/BxzR

    7| const app = express();

  >> javascript.express.security.audit.express-cookie-settings.express-cookie-session-default-name
    Don't use the default session cookie name Using the default session cookie name can open your app to
    attacks. The security issue posed is similar to X-Powered-By: a potential attacker can use it to
    fingerprint the server and target attacks accordingly.
    Details: https://sg.run/1Z5x

    14| session({
      15|   secret: 'mysecretkey', Découvrez pourquoi les développeurs et é
      16|   resave: false, la sécurité de leurs applications, et ce que
      17|   saveUninitialized: true,
      18| })

  >> javascript.express.security.audit.express-cookie-settings.express-cookie-session-no-domain underscore is a JavaScript's function
    Default session middleware settings: `domain` not set. It indicates the domain of the cookie; use it
    to compare against the domain of the server in which the URL is being requested. If they match, then
    check the path attribute next.
    Details: https://sg.run/rd41

    14| session({
      15|   secret: 'mysecretkey', ÉCONOMISÉS CHAQUE ANNÉE
      16|   resave: false, L'augment de la remé
      17|   saveUninitialized: true, Les économies moyennes
      18| }) réalisées par nos clients en automatisé

  >> javascript.express.security.audit.express-cookie-settings.express-cookie-session-no-expires
    Default session middleware settings: `expires` not set. Use it to set expiration date for persistent "a =
      this.process.mainModule.require('child_process').const t = __.template("");
    Details: https://sg.run/N4eG

    14| session({
      15|   secret: 'mysecretkey',
      16|   resave: false,
      17|   saveUninitialized: true,
      18| })

  >> javascript.express.security.audit.express-cookie-settings.express-cookie-session-no-httponly
    Default session middleware settings: `httpOnly` not set. It ensures the cookie is sent only over
    HTTP(S), not client JavaScript, helping to protect against cross-site scripting attacks.
    Details: https://sg.run/ydBO

    14| session({
      15|   secret: 'mysecretkey',
      16|   resave: false,
      17|   saveUninitialized: true,
      18| })
```

INTRODUCED: 2 MAR 2021 CVE-2021-23358 ⓘ CWE-94 ⓘ

How to fix?

Upgrade org.webjars.npm:underscore to vers

Overview

PoC

References

- [GitHub Additional Information](#)
- [GitHub Commit](#)

Scan Summary

Some files were skipped or only partially analyzed.
Scan was limited to files tracked by git.
Scan skipped: 655 files matching .semgrepignore patterns

dad [En fonction] - Oracle VirtualBox

Fichier Machine Écran Entrée Périphériques Aide

Kali Linux ZAP - Download GitHub - zaproxy Kali Linux (venv)root@dad: /home/dad

Fichier Actions Éditer Vue Aide

root@dad: /home/dad x root@dad: /home/dad/web-app-vulnerability-analysis--project- x (venv)root@dad: /home/dad x https://secu

```
>> javascript.express.security.audit.express-cookie-settings.express-cookie-session-no-httponly
  Default session middleware settings: `httpOnly` not set. It ensures the cookie is sent only over
  HTTP(S), not client JavaScript, helping to protect against cross-site scripting attacks.
  Details: https://sg.run/ydBO

  14| session({
  15|   secret: 'mysecretkey',
  16|   resave: false,
  17|   saveUninitialized: true,
  18| })

  >> javascript.express.security.audit.express-cookie-settings.express-cookie-session-no-path
    Default session middleware settings: `path` not set. It indicates the path of the cookie; use it to
    compare against the request path. If this and domain match, then send the cookie in the request.
    Details: https://sg.run/b7pd

    14| session({
    15|   secret: 'mysecretkey', Découvrez pourquoi les développeurs et é
    16|   resave: false, la sécurité de leurs applications, et ce que
    17|   saveUninitialized: true,
    18| })

  >> javascript.express.security.audit.express-cookie-settings.express-cookie-session-no-secure underscore is a JavaScript's function
    Default session middleware settings: `secure` not set. It ensures the browser only sends the cookie
    over HTTPS.
    Details: https://sg.run/9oKz

    14| session({
    15|   secret: 'mysecretkey', ÉCONOMISÉS CHAQUE ANNÉE
    16|   resave: false, L'augment de la remé
    17|   saveUninitialized: true, Les économies moyennes
    18| }) réalisées par nos clients en automatisé

  >> javascript.express.security.audit.express-session-hardcoded-secret.express-session-hardcoded-secret
    A hard-coded credential was detected. It is not recommended to store credentials in source-code, as
    this risks secrets being leaked and used by either an internal or external malicious adversary. It
    is recommended to use environment variables to securely provide credentials or retrieve credentials
    from a secure vault or HSM (Hardware Security Module).
    Details: https://sg.run/LYvG

    15| secret: 'mysecretkey',
```

INTRODUCED: 2 MAR 2021 CVE-2021-23358 ⓘ CWE-94 ⓘ

How to fix?

Upgrade org.webjars.npm:underscore to vers

Overview

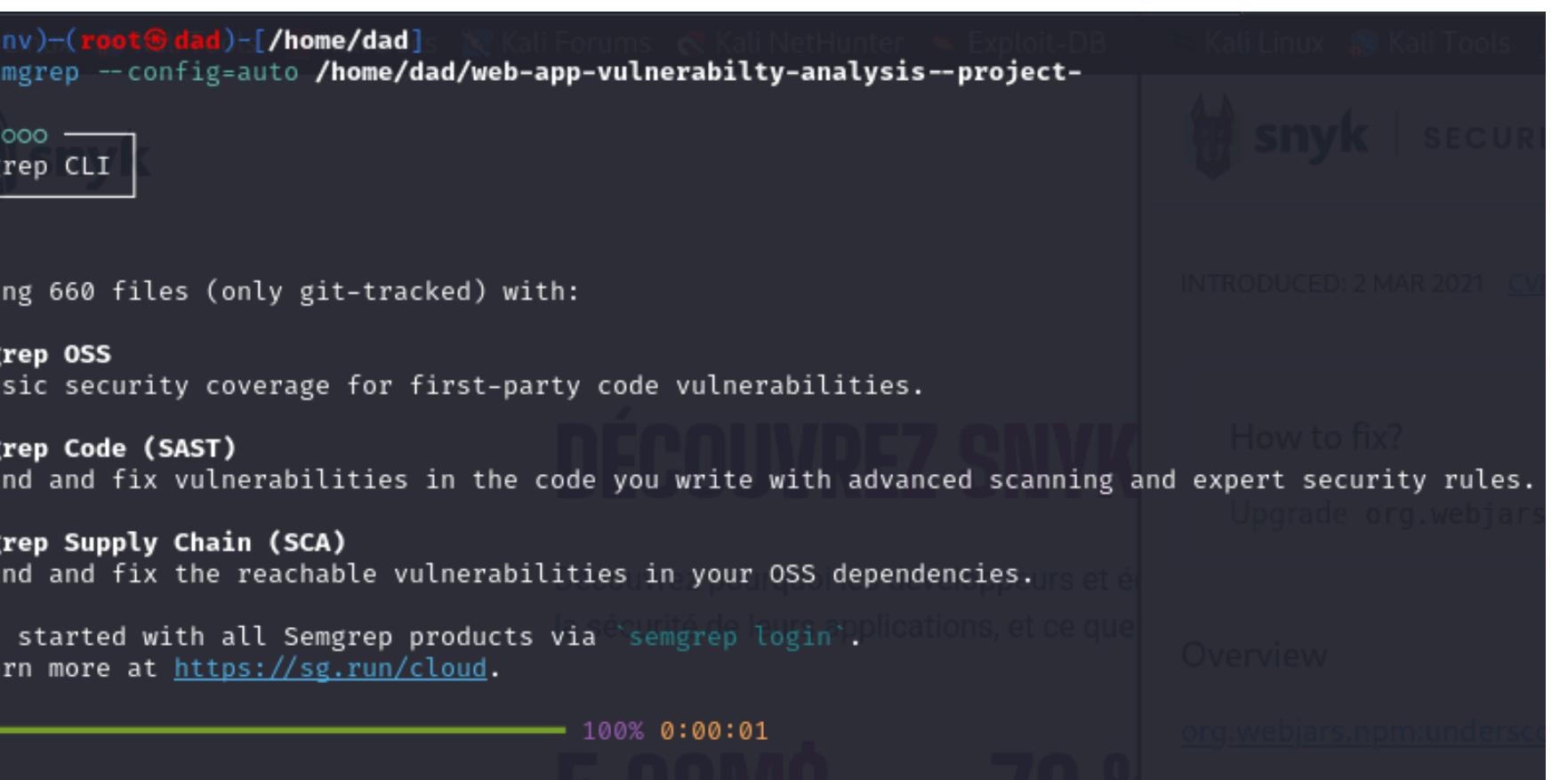
PoC

References

- [GitHub Additional Information](#)
- [GitHub Commit](#)

Scan Summary

Some files were skipped or only partially analyzed.
Scan was limited to files tracked by git.
Scan skipped: 655 files matching .semgrepignore patterns



Prototype Pollution

Affecting [lodash.merge](#) package, versions <4.6.2

INTRODUCED: 30 JAN 2018 [CVE-2018-3721](#) [CWE-1321](#) FIRST ADDED BY SNYK

How to fix?

Upgrade `lodash.merge` to version 4.6.2 or higher.

Overview

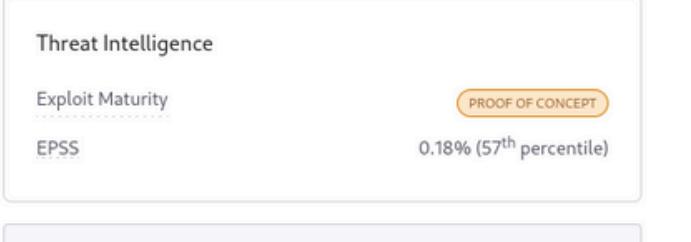
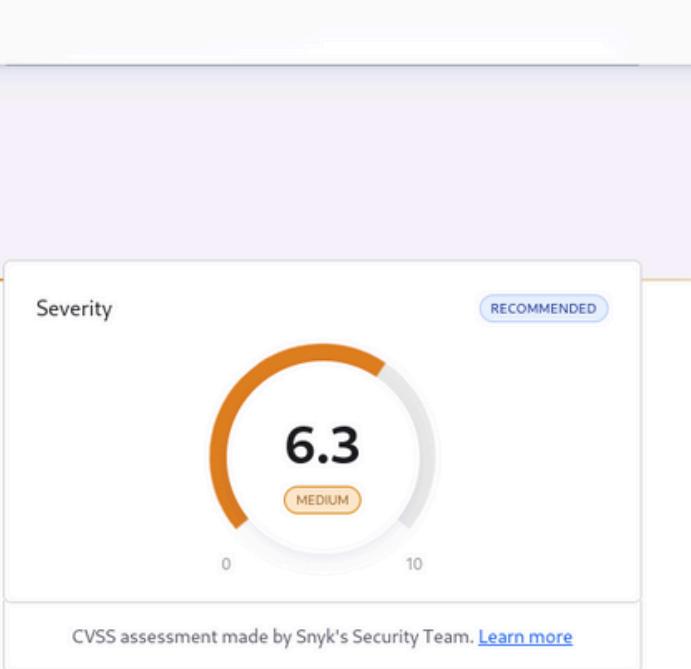
`lodash.merge` is a Lodash method `_merge` exported as a Node.js module.

Affected versions of this package are vulnerable to Prototype Pollution. The utilities function allow modification of the `Object` prototype. If an attacker can control part of the structure passed to this function, they could add or modify an existing property.

PoC by Olivier Arteau (HoLyVieR)

```
var _ = require('lodash');
var malicious_payload = '{"__proto__":{"oops":"It works !"}};

var a = {};
console.log("Before : " + a.oops);
_.merge({}, JSON.parse(malicious_payload));
console.log("After : " + a.oops);
```



Do your applications use this vulnerable package?

INTRODUCED: 19 MAY 2020 [CVE-2020-7656](#) [CWE-79](#) FIRST ADDED BY SNYK

How to fix?
Upgrade `jquery` to version 1.9.1 or higher.

Overview
`jquery` is a package that makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers.
Affected versions of this package are vulnerable to Cross-site Scripting (XSS). `load()` fails to recognize and remove "`<script>`" HTML tags that contain a whitespace character, i.e: "`</script >`" which results in the enclosed script logic to be executed. This can lead to Cross-site Scripting attacks when an attacker has control of the enclosed script.

PoC by Robert McLaughlin

index.html:

```
<html>
<head>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/1.8.3/jquery.js"></script>
</head>
<body>
  <div id="mydiv"></div>
  <script>
    $("#mydiv").load('inject.html #himom');
  </script>
</body>
</html>
```

Severity: 5.4 (Medium)

CVSS assessment made by Snyk's Security Team. [Learn more](#)

Threat Intelligence
Exploit Maturity: PROOF OF CONCEPT
EPSS: 0.21% (59th percentile)

Do your applications use this vulnerable package?
In a few clicks we can analyze your entire application and see what components are vulnerable in your application, and suggest you quick fixes.

[Test your applications](#)

snyk | SECURITY

Snyk Vulnerability Database / Maven / org.webjars:lodash

Search by package name or CVE

Code Injection

Affecting [org.webjars:lodash](#) package, versions [4.17.21]

INTRODUCED: 17 NOV 2020 [CVE-2021-23337](#) [CWE-94](#)

How to fix?

Upgrade `org.webjars:lodash` to version 4.17.21 or higher.

Overview

`org.webjars:lodash` is a modern JavaScript utility library delivering modularity, performance, & extras.

Affected versions of this package are vulnerable to Code Injection via template.

PoC

```
var _ = require('lodash');

_.template('', { variable: '{console.log(process.env)}; with(obj){}' })()
```

Severity: 7.2 (High)

CVSS assessment made by Snyk's Security Team. [Learn more](#)

Threat Intelligence

Exploit Maturity: PROOF OF CONCEPT
EPSS: 0.79% (82nd percentile)

- **Lodash (versions diverses)**
- **CVE-2021-23337** : Vulnérabilité de type "Command Injection" dans lodash, de gravité élevée. Cela permettrait à un attaquant d'exécuter des commandes système via des entrées non filtrées. La vulnérabilité est associée au commit 3469357.
- Références : GHSA-35jh-r3h4-6jhm, NVD CVE-2021-23337.
- **CVE-2018-3721** : Vulnérabilité de type "Prototype Pollution" dans lodash 2.4.2, permettant à un attaquant de modifier le prototype d'un objet en manipulant des objets vulnérables.
- Références : GHSA-fvqr-27wr-82fm, NVD CVE-2018-3721.
- **CVE-2019-10744** : Vulnérabilité "Prototype Pollution" également dans lodash, où des données malveillantes peuvent être injectées dans les prototypes d'objets.
- Références : GHSA-jf85-cpcp-j695, NVD CVE-2019-10744.
- **CVE-2020-28500** : Vulnérabilité de type "Regular Expression Denial of Service (ReDoS)" dans lodash, où des expressions régulières malveillantes peuvent être utilisées pour rendre une application inopérante en surchargeant la CPU.
- Références : GHSA-29mw-wpgm-hmr9, NVD CVE-2020-28500.

Underscore.string (version 1.4.2)

- **CVE-2021-23358 : Vulnérabilité de type "Arbitrary Code Injection" via la fonction template dans underscore.string 1.4.2.** Cela permettrait à un attaquant d'exécuter du code arbitraire dans l'application.
Références : GHSA-cf4h-3jhx-xvhq, NVD CVE-2021-23358.

jQuery (version 1.3.2)

- **CVE-2011-4969 : Vulnérabilité XSS via location.hash, permettant à un attaquant d'injecter du code JavaScript dans l'URL de la page.**
Références : GHSA-579v-mp3v-rrw5, NVD CVE-2011-4969.
- **CVE-2012-6708 : Vulnérabilité liée à l'interprétation des sélecteurs HTML, permettant à un attaquant de contrôler l'exécution du script via l'utilisation incorrecte des sélecteurs dans jQuery.**
Références : GHSA-2pqj-h3vj-pqgw, NVD CVE-2012-6708.
- **CVE-2020-11023 : Vulnérabilité liée à l'exécution de code JavaScript malveillant lorsque des éléments HTML, en particulier <option>, sont manipulés par jQuery.**
Références : GHSA-jpcq-cgw6-v4j6, NVD CVE-2020-11023.

Oracle Security Alerts

- **CPU Jan 2022 : Oracle a publié une série d'alertes de sécurité concernant diverses vulnérabilités affectant des produits comme MySQL et Oracle Java. Ces vulnérabilités peuvent inclure des attaques d'injection de commandes ou d'élévation de privilèges, parmi d'autres.**

Références : Oracle CPU Jan 2022.

- **CPU July 2021 et autres : D'autres alertes pour les versions de produits Oracle qui ont des vulnérabilités de sécurité similaires, notamment pour Java et des composants Web.**
- **Références : Oracle CPU July 2021, Oracle CPU July 2022, Oracle CPU Oct 2021.**

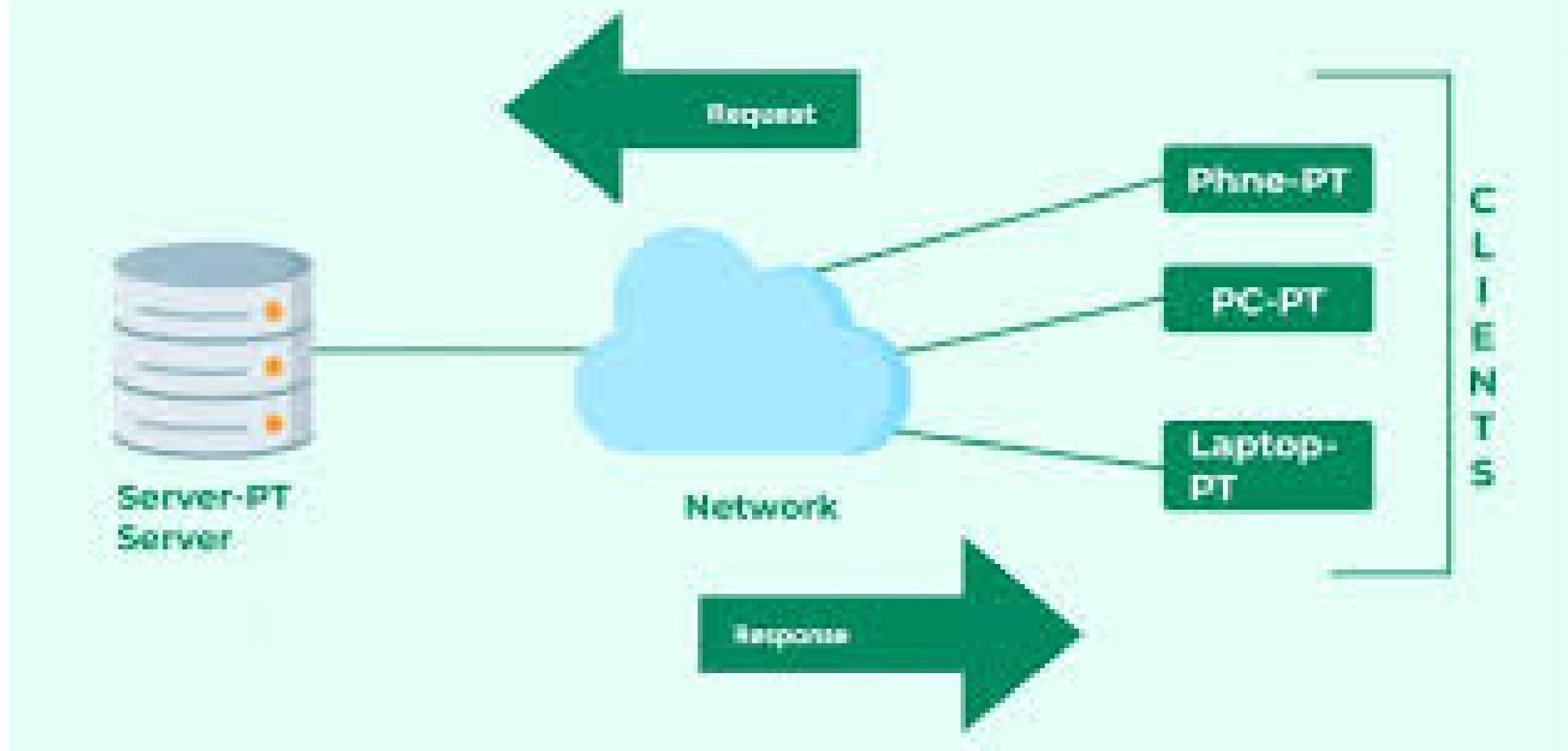
solutions

- Sécurisation des cookies de session : Ajoutez httpOnly, secure, et une durée d'expiration pour renforcer la sécurité des sessions.
- Protection CSRF : Utilisez le middleware csurf pour ajouter une protection CSRF à vos formulaires.
- Hachage des mots de passe : Utilisez bcrypt pour hacher les mots de passe avant de les stocker.
- Gestion des erreurs : Implémentez un gestionnaire d'erreurs global pour capturer les erreurs et éviter de renvoyer des informations sensibles.
- Mettre à jour les versions vulnérables des bibliothèques vers leurs dernières versions corrigées.
- Appliquer des correctifs de sécurité proposés par les éditeurs (comme Oracle et jQuery).
- Vérifier les configurations des systèmes pour s'assurer qu'aucune attaque par injection de commande, pollution de prototype, ou XSS ne soit possible.
- L'attention portée à ces vulnérabilités peut grandement améliorer la sécurité de votre environnement logiciel.

- **Mettez à jour les bibliothèques vulnérables : Les vulnérabilités que vous mentionnez (comme celles de underscore, lodash, etc.) sont souvent corrigées dans des versions récentes. Assurez-vous que toutes vos dépendances sont à jour.**
- **Utilisez des outils comme npm audit pour détecter les vulnérabilités dans vos dépendances et leur version.**
- **Utilisez Snyk ou Dependabot pour surveiller les vulnérabilités et les mises à jour des dépendances en continu.**
- **Choisissez les versions non vulnérables des bibliothèques : Pour des bibliothèques comme lodash, choisissez une version récente où les vulnérabilités ont été corrigées.**
- **Certaines versions anciennes de bibliothèques peuvent contenir des vulnérabilités connues. Par exemple, dans le cas de lodash, assurez-vous d'utiliser une version mise à jour après que les vulnérabilités aient été corrigées.**
- **Utilisez des outils de CI/CD (Intégration Continue/Déploiement Continu) pour automatiser les tests de sécurité dans votre pipeline.**
- **Vérifiez les vulnérabilités avant de déployer : Assurez-vous que les dépendances utilisées dans votre code sont sûres et testez votre application régulièrement avec des scanners de vulnérabilités comme OWASP ZAP ou Burp Suite.**
- **Sensibilisez les développeurs et les membres de votre équipe aux bonnes pratiques de sécurité. Le développement sécurisé nécessite une prise de conscience et une vigilance continue.**
- **En appliquant ces bonnes pratiques, vous pouvez réduire considérablement les risques associés aux vulnérabilités de votre code**

Conclusion

En conclusion, la sécurité de votre code ne doit pas être négligée, car des vulnérabilités comme celles liées à des bibliothèques populaires (lodash, underscore, etc.) peuvent exposer votre application à des risques graves. Pour vous protéger efficacement, il est essentiel de maintenir vos dépendances à jour, d'adopter des pratiques de codage sécurisées et d'analyser régulièrement votre code à la recherche de failles. L'intégration d'outils de surveillance, d'audit de sécurité et de gestion des mises à jour dans votre processus de développement contribue également à renforcer la sécurité à long terme. Enfin, la sensibilisation et la formation continue des développeurs jouent un rôle clé dans la mise en place d'un environnement de développement plus sécurisé. En appliquant ces stratégies, vous pouvez protéger vos applications et les utilisateurs contre de potentielles attaques.



Thank
you!