Projet - Application web vulnérable

Objectifs

- Comprendre les vulnérabilités courantes en web et pouvoir les identifier correctement
- Identifier et exploiter des vulnérabilités web
- Mettre en oeuvre des correctifs afin de corriger les vulnérabilités
- Être capable d'effectuer un audit technique sur une application web
- Travail en équipe

Énoncé

Le but de ce projet est d'auditer une application web volontairement vulnérable afin d'identifier ses faiblesses, de les corriger et d'en faire un rapport.
Cela permet de montrer une bonne compréhension de celles-ci et de les documenter.

Il faudra fournir ensuite une autre application web qui se base sur la première (avec les vulnérabilités) mais qui patch ces vulnérabilités précédemment mises en place afin qu'elle soit totalement sécurisée.

Vous avez le choix entre **DEUX** applications qui sont presque identiques (à peu de choses près) :

- En PHP (Symfony) : https://drive.google.com/file/d/1y8VJLw3U3U8ffPffyghwgy9h9RGPH0qU/view?usp=sharing
- En JS : https://drive.google.com/file/d/1FCYDOAEYIHyzCAimsrDCYAkaBOJRun_0/view?usp=sharing

Vous pouvez choisir celle que vous souhaitez parmi les deux. Il n'est PAS OBLIGATOIRE de faire les deux. Faites en fonction de vos affinités avec les technos. Cependant, si vous souhaitez sécuriser les deux applications, ce ne sera que bénéfique pour vous.

Groupe de 2 personnes max. Projet à rendre avant le **mercredi 7 janvier** à **13h00** (à l'issue de la dernière heure de cours).

Des séances sont prévues ensemble pour que vous puissiez me poser des questions et que je vous aide au niveau de la mise en place/détection/remédiation des vulnérabilités. Vous pouvez cependant prendre de l'avance de votre côté.

Installation applications

PHP

Un README.txt est présent dans chaque application afin que vous puissiez suivre les instructions.
Je le remets ici.

Lancer :

```
docker compose build
docker compose up -d
```

Il faut après remplir la base de données, pour cela, entrer dans l'image `vulnerable-symfony-php` en lançant :

```
docker compose exec -it vulnerable-symfony-php bash
```

Une fois dans l'image, copier le paragraphe complet des commandes suivantes qui vont se lancer en une fois :

```
composer require --dev doctrine/doctrine-fixtures-bundle &&
composer install &&
php bin/console d:d:c &&
php bin/console d:s:u --force &&
php bin/console d:f:l &&
npm i &&
npm run dev
```

Écrire `yes` quand c'est demandé.

Quitter le conteneur avec `exit`.

1. Vous pouvez créer un compte sur chaque application :
   - http://localhost/

JS

```
docker compose up --build -d
```

- Le frontend React s'exécute sur le port 4000 http://localhost:4000
- Le backend ExpressJs s'exécute sur le port 4001 http://localhost:4001
- La base de donnée mysql s'exécute sur le port 4002
- Phpmyadmin est accessible via le port 4003 http://localhost:4003

Livrables

Au final, vous devrez rendre deux éléments :

- Une application web sécurisée (qui corrige les vulnérabilités)
- Un rapport

Le rapport doit permettre de savoir quelles sont les vulnérabilités que vous avez identifié, un PoC (Proof of Concept) de leur exploitation (avec des captures + explications), les remédiations associées et un "contre-audit" pour prouver qu'elle n'est plus exploitable.
Si vous utilisez des outils et/ou scripts externes, n'oubliez pas de mettre un lien vers ceux-ci dans le rapport (Github, lien de téléchargement ou autre).

Attention cependant, si d'autres vulnérabilités sont trouvées (lors de l'évaluation) ou que le patch n'est pas suffisant et qu'il peut être bypass, vous perdrez des points.

Le projet doit être rendu **AU MAXIMUM mercredi 7 janvier à 13h00** !
Tout retard sera sanctionné au niveau de la note.

N'oubliez pas de mettre le nom de toutes les personnes du groupe sur votre rapport.

Critères d'évaluation

Vous serez évalué sur différents aspects du projet :

- Compréhension des vulnérabilités et capacité à les identifier / reproduire : **7 pts**
- Qualité et cohérence des patchs mis en place pour sécuriser l'application : **6.5 pts**
- Exhaustivité, explications et structure du rapport : **6.5 pts**

Conseils - Détails

Voici quelques conseils pour vous permettre de mener à bien ce projet :

- Il vous est recommandé de commencer d'abord votre test en *black box*. C'est à dire comme un utilisateur lambda, sans avoir besoin de regarder dans le code source. Une fois que vous avez identifié quelques vulnérabiliés, si vous bloquez sur l'exploitation ou que vous voulez aller plus loin, vous pouvez aller voir le code source (dans tous les cas vous y serez obligé pour corriger la vulnérabilité).
- Rapport sous format PDF de préférence
- Partage du dossier projet sur un Github ou un Drive (Google, DropBox, Mega...) : attention cependant à bien configurer les droits d'accès pour que je puisse y accéder via un lien par exemple pour l'évaluation et la correction.
- Pour le rapport, imagez avec des captures d'écran et expliquez vos raisonnements (pourquoi vous avez fait telle ou telle chose).
- Pour le code, pensez à bien le documenter/commenter pour faciliter la correction.
- Une application non fonctionnelle retirera forcément des points. Si vous la corrigez mais qu'elle ne fonctionne plus ou que vous avez retiré la fonctionnalité, ce n'est pas bon.
- Pour la démonstration de l'exploitation des vulnérabilités, vous pouvez utiliser des outils vus en cours (Burp Suite, ffuf, etc...), d'autres outils dont on n'a pas forcément parlé (SonarQube, etc..) ou bien les exploiter manuellement. Dans tous les cas, il faut expliquer pourquoi telle ou telle chose fonctionne, comment et détailler le processus d'exploitation (ça ne sert à rien de juste lancer SQLmap pour prouver qu'il y a une SQLi s'il n'y a aucune explication de la faille derrière).

Je reste disponible même en dehors des cours si vous avez des questions : quentin.simier@synoslabs.com