Remote Cookies Stealing

SIWAR JENHANI (RT4)
SOUHIR FARES (RT4)

Sommaire :

Contenu

I. Introduction:

Tout ordinateur connecté à un réseau informatique est potentiellement vulnérable à une attaque si aucune défense n'a été mise en place.

Une « **attaque** » est l'exploitation d'une faille d'un système informatique à des fins non connues par l'exploitant du système et généralement préjudiciables.
Afin de contrer ces attaques il est indispensable de connaître les principaux types d'attaques afin de mettre en œuvre des dispositions préventives.

Les motivations des attaques peuvent être de différentes sortes :

- voler des informations, tels que des secrets industriels ou des propriétés intellectuelles ;
- glaner des informations personnelles sur un utilisateur ;
- récupérer des données bancaires ;
- s'informer sur l'organisation (entreprise de l'utilisateur, etc.) ;
- troubler le bon fonctionnement d'un service ;

Parmi les différents types d'attaque les plus connu, on citera l'attaque de type **Cross-Site Scripting** (notée parfois *XSS* ou *CSS*).

II. Présentation de l'atelier :

Les attaques de type **Cross-Site Scripting**, abrégé XSS (Cross voulant dire « croix » en anglais. Le symbole X a été choisi pour le représenter), sont des attaques visant les sites web affichant dynamiquement du contenu utilisateur sans effectuer de contrôle et d'encodage des informations saisies par les utilisateurs. Les attaques Cross-Site Scripting consistent ainsi à forcer un site web à afficher du code HTML ou des scripts saisis par les utilisateurs.

C'est une faille ou nous pouvons injecter du code JavaScript pour voler des cookies, en parti - culier celui du webmaster qui est intéressant afin que le site web nous reconnaisse en tant qu'administrateur.

Cette faille est dite "côté client" puisque notre code (x)HTML et JavaScript n'est pas parsé par le serveur, qui lui se contente de renvoyer la page, mais bien interprété par le navigateur de la victime.

Un cookie ?

Un cookie en informatique c'est un petit bloc de données sous forme d'un fichier texte forgé par un site web et récupéré à la prochaine connexion, il est utilisé pour stocker une information spécifique sur l'utilisateur.

Il y'a deux types d'attaques :

1) Attaque persistante :

- L'attaque est dite « **persistante** » lorsque les données saisies par l'utilisateur sont stockées sur le serveur pendant un certain temps ,cas d'un forum de discussion par exemple : Chaque message est enregistré pour être ensuite affiché lorsqu'un utilisateur demande la page, imaginons que l'on insère du code malicieux dans un de nos messages, ce dernier serait donc affiché à chaque demande d'un utilisateur et chaque utilisateur le consultant serait alors à la merci du code malicieux ! On pourrait alors piéger tous les utilisateurs

2) Attaque non persistante :

- Les attaques dites « **non persistantes** » concernent les pages web ici, les informations ne sont stockées nulle part, le serveur utilise les données envoyées par l'utilisateur pour fournir une page dynamique en résultant. En effet il faut confectionner un lien pointant vers la page faillible, puis faire en sorte que la victime clique dessus. L'exemple de faille XSS fournit plus haut présente une faille de ce type, pour l'exploiter il faudrait donc faire cliquer la victime sur un lien de ce type : "page.php?pseudo=<script>blablabla</script>" qui consultera alors une page modifiée, piégée.

Cet atelier a pour objectif de simuler une attaque non persistante sur un forum vulnérable à la faille XSS.

III. Présentation des outils utilisés :

1) Acunetix Web Vulnerability Scanner

Acunetix Web Vulnerability Scanner est un logiciel qui permet d'évaluer le niveau de sécurité de votre site web. En effet, un site web peut être la cible des pirates surtout s'il possède des informations sensibles sur ses utilisateurs tels que les numéros de carte bancaires. Le logiciel teste la vulnérabilité de votre site face aux injections SQL et le cross Scripting. Ce sont des techniques très répandues pour pirater un site web afin d'en récolter des informations soit directement dans la base de données, les cookies, ou autres endroits susceptibles de contenir des informations confidentielles.
N.B : C'est un logiciel payant.

2)

3) WampServer :

WampServer (anciennement WAMP5) est une plateforme de développement Web de type WAMP, permettant de faire fonctionner localement (sans se connecter à un serveur externe) des scripts PHP. WampServer n'est pas en soi un logiciel, mais un environnement comprenant deux serveurs (Apache et MySQL), un interpréteur de script (PHP), ainsi que phpMyAdmin pour l'administration Web des bases MySQL.
N.B : C'est un logiciel libre.

On peut utiliser notamment EasyPHP

WampServer

4) Notepad++ :

Notepad++ est un programme spécialement conçu pour l'édition de code source. Il est compatible avec plusieurs langages de programmation.
N.B : c'est un logiciel libre

5) Add-on greasemonkey :

Greasemonkey est une extension pour le navigateur web Mozilla Firefox permettant de modifier le comportement d'une page web en associant un script JavaScript au chargement de celle-ci.

6) Add-on cookie Injector :

Le script utilisateur CookieInjector lui permet d'injecter les cookies de la page Web en cours de consultation.

7) Add-on Edit this cookie:

C'est une extension pour le navigateur Google chrome, Il permet d'ajouter, supprimer, modifier, rechercher, protéger et bloquer les cookies

VI. Topologie :

1) Création d'un forum :

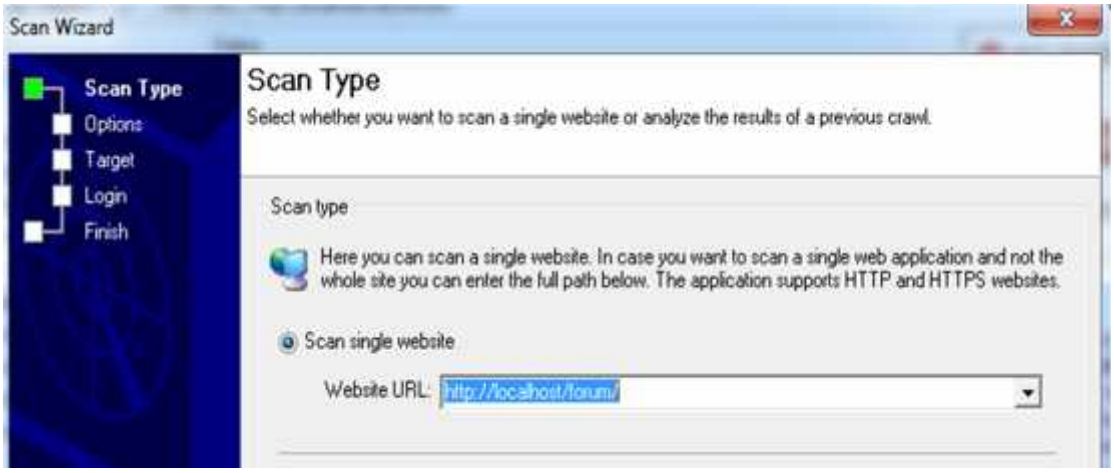
On va créer un forum simple vulnérable à la faille XSS, qui contient une page d'accueil afin d'accéder au forum et une page de commentaire :

Se connecter en tant que administrateur ou bien utilisateur :

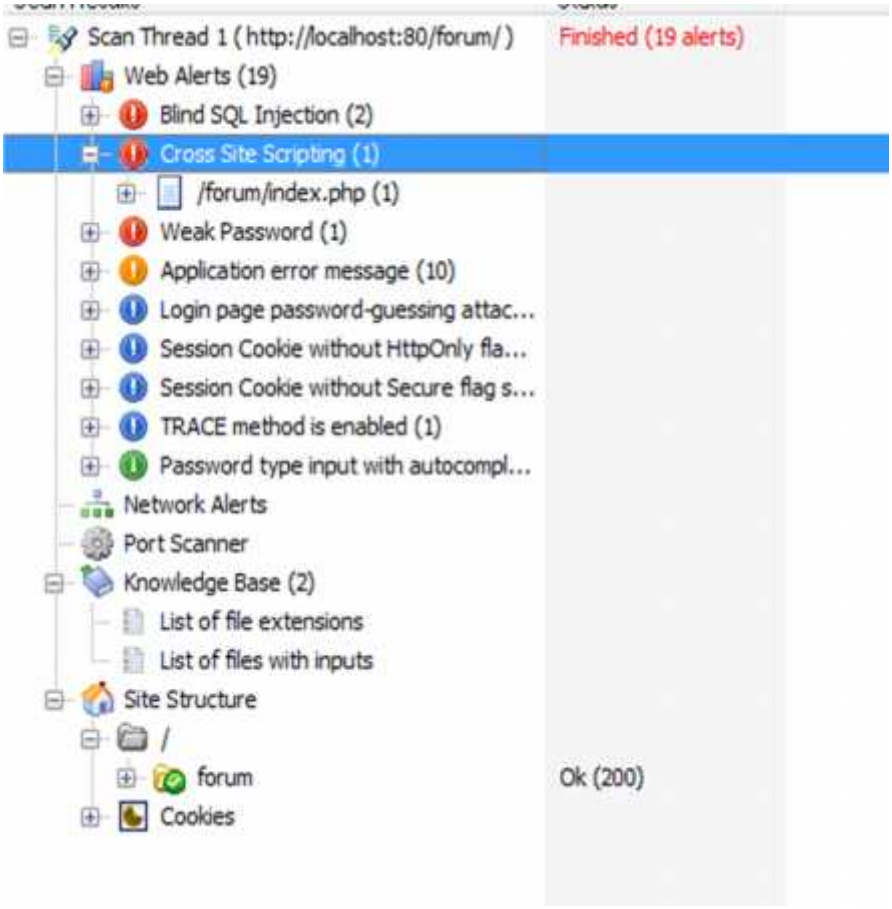
2) Scanner la vulnérabilité du forum :

On va scanner le forum à l'aide d'Acunetix Web Vulnerability Scanner afin d'analyser les vulnérabilités de forum et précisément la vulnérabilité Cross-Site Scripting (XSS).

L'URL de notre forum est http://localhost/forum/



On lance le scan :



On remarque bien que notre forum contient une faille XSS.

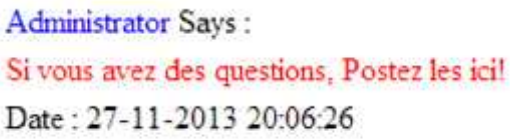
IV. Scénario de test :

On va connecter en tant que administrateur et utilisateur depuis notre machine comme nous travaillons en local.

Se connecter en tant que administrateur et utilisateur dans une autre fenêtre de navigation :

L'administrateur poste un message de bienvenue

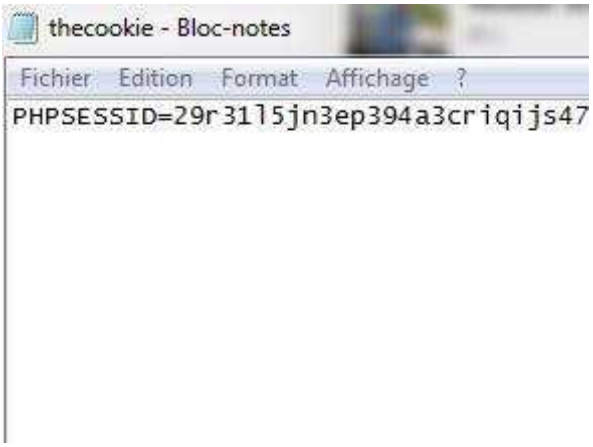
Il demande aux utilisateurs de poster leurs questions sur le Forum, s'ils ont confrontés des problèmes



Un utilisateur qui va jouer le rôle d'un Hacker va poster son message en lui ajoutant un script qui fait appel à une fonction dans le but de voler les cookies de l'administrateur.

Le message va être posté à l'administrateur.

Le script de hacker consiste à enregistrer les cookies de l'administrateur dans un fichier texte « cookie.txt ». Il va consulte son fichier après l'envoi de script à l'administrateur



Après avoir récupérer les cookies, la hacker va utiliser le add-on Cookie injector pour injecter le cookie de l'administrateur afin d'obtenir les droits d'accès du webmaster:

En cliquant sur alt+c la fenêtre de Cookie injector sera affichée qui contient une zone de texte qu'on va écrire le code de cookie qu'on a récupérer dedans.

Après avoir actualisé sa page, le hacker qui est connecté en tant que utilisateur sera connecté au forum en tant que Administrateur.

A ce moment-là, le hacker accède au forum en tant que administrateur avec tous ses privilèges, il peut donc :

- Changer le mot de passe de l'administrateur afin que ce dernier ne puisse pas accéder à son forum.
- Accéder à la base des données du forum.
- Récupérer les informations personnelles des utilisateurs.
- Faire circuler des fausses informations.

V. Protection :

Du côté de l'utilisateur, il est possible de se prémunir contre les attaques CSS en configurant le navigateur de manière à interdire l'utilisation de la balise html <script>. Mais cela n'est pas suffisant, car on peut contourner facilement la protection en utilisant d'autres balises html, comme la balise iframe, Il est donc plus prudent d'interdire toute utilisation de balises html. La fonction **Php strip_tags()** permet de faire cela et peut aller plus loin en autorisant seulement certaines balises mais là encore, cette méthode est insuffisante face aux exploits n'utilisant aucune balise.

Dans la réalité cette solution est souvent trop contraignante pour l'utilisateur car de nombreux sites refuseront de fonctionner correctement en l'absence de possibilité d'exécution de code dynamique.

La seule façon viable d'empêcher les attaques Cross-Site Scripting consiste à concevoir des sites web non vulnérables. Pour ce faire, le concepteur d'un site web doit :

- Vérifier le format des données entrées par les utilisateurs
- Encoder les données utilisateurs affichées en remplaçant les caractères spéciaux par leurs équivalents HTML en utilisant l'une des deux méthodes : **htmlspecialchars(),htmlentities()**

htmlentities() c'est du superflu, il encodera entité html tous les caractères (même un simple a)
htmlspecialchars() n'encode que les caractères potentiellement dangereux, donc ici on a moins de transformation = moins de ressource consommé .

VI. Conclusion :

En conclusion, et après tout ce que nous venons de voir, nous pouvons dire que les failles XSS ne sont pas des failles à négliger étant donné qu'elles permettent, par une attaque bien menée, d'effectuer n'importe quelles actions sur le site faillible. Et malgré cela, ces failles restent parmi les plus présentes sur Internet ! Il faut donc informer les Webmasters du danger potentiel qu'instaure une XSS sur un site. Conseils de sécurité : c'est que la base de la base c'est de partir du principe que tout ce qui vient du visiteur (ou de son navigateur) est pirate et qu'il faut donc se protéger.