Sommaire

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 : B2evolution Beta 4.0.0.1
Description : CMS pour multi blogs
URL : http://b2evolution.net
Date : Mercredi 24 novembre 2010

4 Les participants

kr0ch0u

K1wy

Essandre

Tishrom.

5 Compte rendu de l'audit

Fichier : gettext/xg.php

Ligne : 113

```
elseif( isset($argv[2]) && strtoupper($argv[2]) == 'MERGE' )
{
    $action = 'merge';
    if( ! isset($argv[3]) ) // the to-get-merged locale
    {
         echo_usage();
         exit(1);
    }
    $locales_to_merge = array_slice( $argv, 3 );
}
```

Ligne : 254

```
if( $action == 'merge' )
{ // Merge with existing .po files:
     if( ! @is_readable( $file_pot ) )
     {
          echo "FATAL: $file_pot is not readable!\n";
          exit(1);
     }
     foreach( $locales_to_merge as $l_locale )
     {
          $l_file_po = $dir_root.'locales/'.
$l_locale.'/LC_MESSAGES/messages.po';
          echo 'Merging with '.$l_locale.'.. ';
          if( ! file_exists( $l_file_po ) )
          {
               echo "PO file $l_file_po not found!\n";
               continue;
          }
          system( 'msgmerge -U -F --no-wrap '.escapeshellarg($l_file_po).'
'.escapeshellarg($file_pot) );
          # delete old TRANS comments and make automatic ones valid comments:
          system( 'sed -i -r "/^#\\s+TRANS:/d; s/^#\\. TRANS:/# TRANS:/;" '.
$l_file_po );
          echo "Written $l_file_po .\n";
          echo "\n";
     }
     exit(0);
}
```

Exécution possible de commande : Si $l_file_po = “\00 && commande #” marche uniquement sous windows pour le null-byte poisoning. Car file_exists($dir_root.'locales/') renvoi true et system('sed -i -r ”/^#\\s+TRANS:/d; s/^#\\. TRANS:/# TRANS:/;” '.$l_file_po); oublie de escapeshellarg.

Ligne : 291

```
$l_file_po = $dir_root.'locales/'.$l_locale.'/LC_MESSAGES/messages.po';
$global_file_path = $dir_root.'locales/'.$l_locale.'/_global.php';
```

Ligne : 304

```
$r = $POFile->write_evo_trans($global_file_path, $l_locale);
```

Fichier : blogs/inc/locales

Ligne : 223

```
function write_evo_trans($file_path, $locale)
```

Ligne : 241

```
fwrite( $fp, '$trans[\''.$locale."'] = array(\n" );
```

Si $l_locale = ”../blogs/index.php\0'] = array(); code_php/*”

Normalement, ca passe les file_exists. A voir si cela passe le $trans[\`.$locale.”'] toujours sous windows.`

Ligne : 52

```
param( 'redirect_to', 'string', );
```

La fonction param() prend en argument des variable $_GET comme $_POST. Il est possible d'agir sur la variable $redirect_to depuis l'url : register.php?redirect_to=[fake_url] et de redirigé le nouveau membre vers une page qui n'a rien à voir avec celle du site de base. Pas pris le temps de chercher dans le fichier les lignes vulnérable au niveau de la non verification de l'url.

Cotés admin :

Fichier : htsrv/async.php:

Ligne : 142

```
http://localhost/b2evolution/blogs/htsrv/async.php?
action=get_login_list&q=admin'%20union%20select%20user()%20--%20a#
```

– > SQL Injection

Fichier : _commentquery.class.php:

Ligne : 270

```
$this->WHERE_and( $this->dbprefix.'author_url '.$url_match.' ("'.
$author_url.'")'.$include_empty );
```

– > PHP_INFO non enlevé à l'install