PHP Security Scanner

1) Informations générales

Rubrique	Outil d'audit de code PHP
Public	Développeurs
Editeur	PHP Security Scanner Development Team
Licence	GPL
Fonctionnalités	Recherche de motifs (fonctions avec variables), failles d'inclusion (code, commandes)

2) Installation

Environ 20 minutes :

- ✓ télécharger securityscanner-1.0.2.tar.gz version 1.0.2 (07 juin 2007) http://sourceforge.net/project/showfiles.php?group_id=141049
- ✓ décompresser l'archive, elle contient :
  - bin = outils d'analyse du code
  - interface = interface de consultation des résultats de l'analyse
  - 3dPartyLib = bibliothèques pour l'interface
- ✓ créer un répertoire security_scanner dans le DocumentRoot et placer dans ce répertoire les répertoires interface et 3dPartyLib.
- ✓ créer la base mysql et placer les données (fichier db_generate.sql) : table problem = liste des motifs à rechercher, table result = résultat de l'analyse
- ✓ mettre les informations de connexion à la base pour l'analyseur et l'interface :
  - copier bin/config.php.sample en config.php et renseigner les variables de connexion ($hostname, $db_username, $db_password, $database_name).
  - copier interface/lib/config.php.sample en interface/lib/config.php et renseigner les variables de connexion
- ✓ tester dans le navigateur la page d'accueil du répertoire interface, elle devrait afficher : "No results in the database. You should use the executable first."

3) Utilisation

3.1) Lancer une analyse

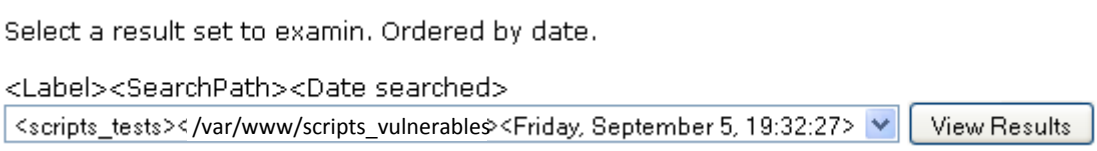
Exécuter le script security_scan.php du répertoire bin avec PHP en ligne de commande :

```
php security_scan.php repertoire_scripts etiquette
```

3.2) Analyser les résultats

Le scanner recherche des motifs contenant des **fonctions** potentiellement dangereuses qui utilisent des **variables**. Les fonctions sont définies dans la table problem de la base mysql : exec, passthru, proc_open, shell_exec, system, popen, pcntl_fork, pcntl_exec, fsockopen, pfsockopen, socket_bind, sockent_accept, socket_listen, socket_create, stream_socket_client, stream_socket_server, dl, include, include_once, require, require_once, fopen, readfile, file, phpinfo, eval…

Entrer l'URL du répertoire interface. Une liste déroulante propose les résultats d'analyse (<étiquette> <chemin du répertoire><date d'analyse>).



L'interface de consultation interroge la base mysql. Le résultat est affiché dans le navigateur :

Search label: scripts_tests	Search made on: Friday, September 5, 19:32:27
Search path:	/var/www/scripts_vulnerables
Results found: 11	
1	
Back...	
NOTE: Links in Result column are not usable yet. They point the file path to check problem yourself. See TODO file for more details.	

Result #	Line #	Line Contents Problem description
1	8	readfile($_GET['fichier']); Outputs a file. Check if data is properly validated.
2	9	echo "\n1. ", system($cmd); If you are going to allow data coming from user input to be passed to this function, then you should be using escapeshellarg() or escapeshellcmd() to make sure that users cannot trick the system into executing arbitrary commands. Check if data is properly validated.
3	11	echo "\n3. ", system($_GET['cmd']); If you are going to allow data coming from user input to be passed to this function, then you should be using escapeshellarg() or escapeshellcmd() to make sure that users cannot trick the system into executing arbitrary commands. Check if data is properly validated.
4	13	echo "\n4. ", exec($cmd); If you are going to allow data coming from user input to be passed to this function, then you should be using escapeshellarg() or escapeshellcmd() to make sure that users cannot trick the system into executing arbitrary commands. Check if data is properly validated.
5	15	echo "\n6. ", exec($_GET['cmd']); If you are going to allow data coming from user input to be passed to this function, then you should be using escapeshellarg() or escapeshellcmd() to make sure that users cannot trick the system into executing arbitrary commands. Check if data is properly validated.
6	20	echo "\n9. ", shell_exec($cmd); If you are going to allow data coming from user input to be passed to this function, then you should be using escapeshellarg() or escapeshellcmd() to make sure that users cannot trick the system into executing arbitrary commands. Check if data is properly validated.
7	22	echo "\n11. ", shell_exec($_GET['cmd']); If you are going to allow data coming from user input to be passed to this function, then you should be using escapeshellarg() or escapeshellcmd() to make sure that users cannot trick the system into executing arbitrary commands. Check if data is properly validated.
8	8	require($_GET['page']); If "URL fopen wrappers" are enabled in PHP (which they are in the default configuration), you can specify the file to be included using a URL. Check if data is properly validated.
9	10	include($_GET['page']); If "URL fopen wrappers" are enabled in PHP (which they are in the default configuration), you can specify the file to be included using a URL. Check if data is properly validated.
10	12	require_once($_GET['page']); If "URL fopen wrappers" are enabled in PHP (which they are in the default configuration), you can specify the file to be included using a URL. Check if data is properly validated.
11	14	include_once($_GET['page']); If "URL fopen wrappers" are enabled in PHP (which they are in the default configuration), you can specify the file to be included using a URL. Check if data is properly validated.

4) Conclusion

Avantages :

- Ecrit en PHP
- Permet d'identifier rapidement les scripts dans lesquels des **fonctions** potentiellement dangereuses contiennent des **variables** et affiche une explication avec des conseils.
- Détecte les failles d'inclusion

Inconvénients :

- **Faux positifs** : <?php $cmd = 'ls –la' ; system($cmd) ?>
- Ne détecte pas : XSS, opérateur backtick, injections SQL
- Utilise une base de données pour stocker les réponses et les vulnérabilités.

5) Sur Internet

Spike PHP Security Scanner http://securityscanner.lostfiles.de