Local file inclusion in page handling

EXPLICATION

Comme remarqué au cours de la phase d'énumération (voir ENUMERATION – partie 12), l'application web semble gérer l'affichage de ses pages par le biais d'URLs de ce genre :

```
http://192.168.1.37/index.php?page=signin
http://192.168.1.37/index.php?page=survey
...
```

Il est possible qu'en back end, une fonction PHP du type **include** soit utilisée par **index.php** afin d'afficher la bonne page à chaque fois. Le contenu du paramètre d'URL **page** semble donc utilisé dans le cadre de ce mécanisme. Ce dernier peut être modifié par l'utilisateur, qui contrôle donc l'input passé à la fonction du type **include**.

Une telle configuration est dangereuse si l'input utilisateur n'est pas correctement validé, car cela peut tout à fait résulter en une faille de type **Local File Inclusion**. En effet, que se passerait-il si jamais l'utilisateur décidait de fournir à l'application l'URL suivante :

```
http://192.168.1.37/index.php?page=../../../../../../../../../../etc/passwd
```

En backend, PHP pourrait tout à fait exécuter une fonction de ce type :

```
include "../../../../../../../../../../etc/passwd"
```

Ce qui révélerait à l'utilisateur le contenu du fichier `/etc/passwd` (si aucune restriction **open_basedir** n'est en place en tout cas ; sinon, on ne pourrait remonter que jusqu'au dossier spécifié dans cette variable de configuration PHP).

Lorsqu'on fournit à l'application une URL de ce type, on récupère un **flag**.

RESSOURCES

Un script permet d'envoyer cette requête et donc de générer la page de réponse avec le flag dans le dossier Ressources.

MITIGATIONS :
> Éviter au maximum de donner à l'utilisateur la possibilité de contrôler le contenu des fonctions de type include.
> Mettre en place une validation rigoureuse du user input passé à ce type de fonction.
> Mettre en place une restriction de type open_basedir.