Rapport des TP1 et TP2 : Authentication using LDAP and Kerberos Web Application Security

dans le cadre du Master 2 Cybersécurité et e-Santé

Année universitaire 2021 – 2022

Présenté par Quentin GUARDIA, quentin.guardia@etu.u-paris.fr

Sous la direction de Saad EL JAOUHARI

Table des matières

Capture du trafic Kerberos.....2
Challenges OWASP Juice Shop.....3

Capture du trafic Kerberos

No.	Time	Source	Destination	Protocol	Length	Info
63	3.305231443	192.168.8.92	192.168.8.16	KRB5	235	AS-REQ
64	3.306239251	192.168.8.16	192.168.8.92	KRB5	858	AS-REP
65	3.415456552	192.168.8.92	192.168.8.16	KRB5	586	TGS-REQ
66	3.416482568	192.168.8.16	192.168.8.92	KRB5	893	TGS-REP

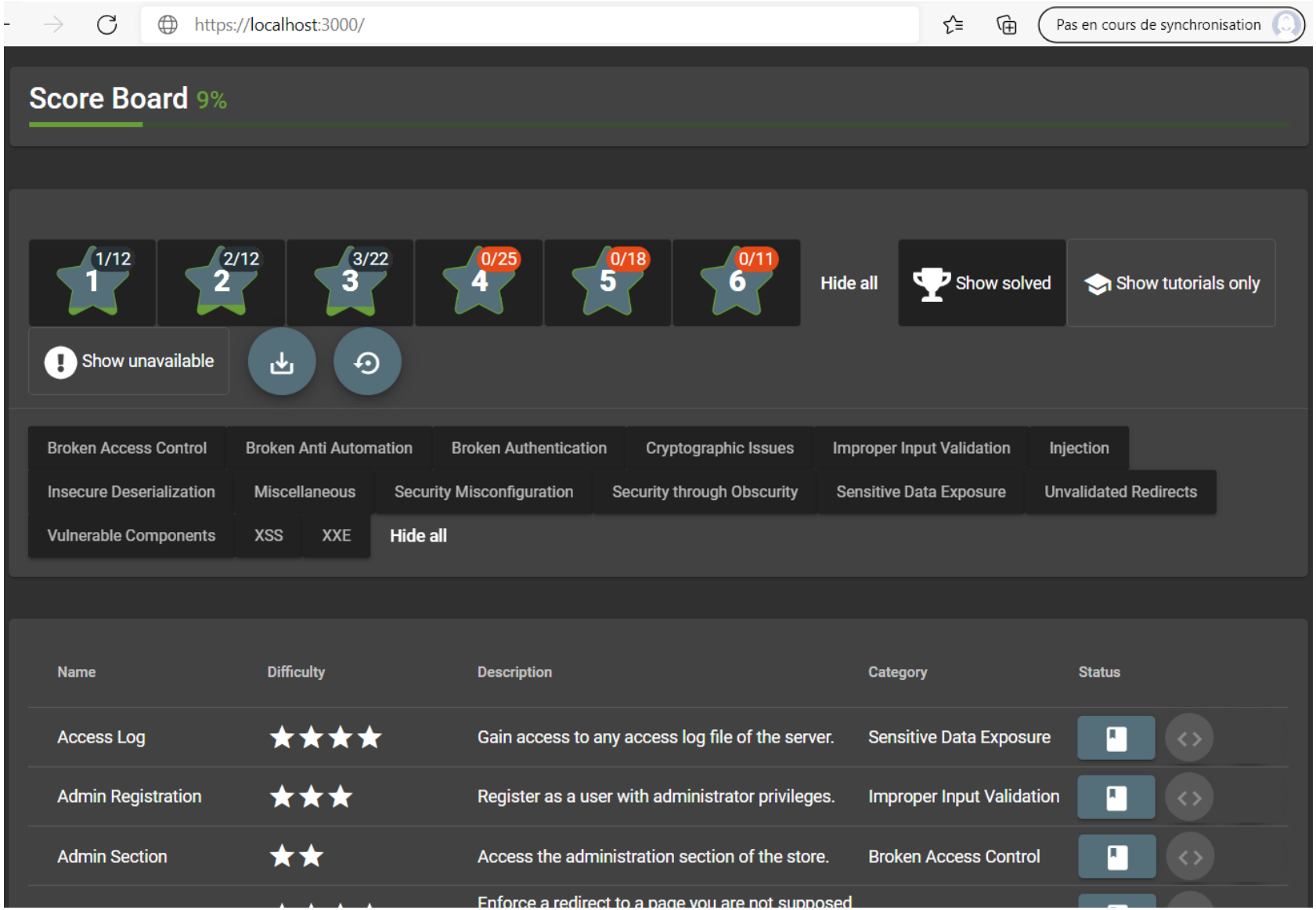
Sur la capture Wireshark ci-dessus, l'adresse IPv4 locale du serveur est 192.168.98.16 et l'adresse du client, 192.168.8.92.

En cherchant sur Wireshark, on a la confirmation que ni l'identifiant, ni le mot de passe ne sont transmis en clair. En effet, Kerberos est un protocole d'authentification qui fonctionne sur la base de tickets pour permettre aux nœuds communiquant sur un réseau non sécurisé de prouver leur identité les uns aux autres de manière sécurisée. Il est basé sur un modèle client-serveur et il assure une authentification mutuelle : l'utilisateur et le serveur vérifient l'identité l'un de l'autre. Les messages du protocole Kerberos sont protégés contre les attaques MITM. Kerberos s'appuie sur la cryptographie à clé symétrique et nécessite un tiers de confiance.

Ici, les tickets apparaissent sous la forme des paquets suivants :

- « AS-REQ » pour la demande d'authentification du client, par l'intermédiaire du contrôleur de domaine,
- « TGS-REP » pour l'acceptation par le serveur,
- « TGS-REQ » pour la demande de ressource de la part du client,
- « TGS-REP » pour fournir au client un ticket avec le nom de service correspondant à la requête précédente, ainsi qu'une clé. Cela via le contrôleur de domaine.

Challenges OWASP Juice Shop



Challenge name	Category	Difficulty	Description	How to solve it ?
Login Admin	SQL Injection	**	Log in with the administrator's user account.	La requête SQL est vulnérable aux apostrophes (‘), il suffit d’entrer dans le champs d’email `‘ OR TRUE --` pour que la requête de connexion soit toujours vraie (et la suite en commentaire) et permette la connexion. On s’aperçoit alors qu’on est connecté en tant qu’administrateur.
Login as Bender	SQL Injection	***	Log in with Bender's user account.	Comme nous l'avons vu, il s’agit ici connecter au compte de Bender. Il faut à nouveau faire une tentative de connexion, mais cette fois nous mettrons : `bender@juice-sh.op'--` comme email.
DOM XSS	XSS	*	Perform a *DOM* XSS attack with : `<iframe src="javascript:alert(`xss`)">.`	Nous utilisons l'élément iframe avec une balise d'alerte javascript : `<iframe src="javascript:alert(xss)">` La saisie de cette information dans la barre de recherche déclenchera l'alerte.
Reflected XSS	XSS	**	Perform a *reflected* XSS attack with : `<iframe src="javascript:alert(`xss`)">`	On se connecte au compte admin et on accède à la page "Order history". À partir de là, on voit une icône "Truck". En cliquant dessus, on accède à la page des résultats de course. On voit également qu'il y a un identifiant associé à la commande. On utilise l'iframe XSS, `<iframe src = "javascript:alert(xss)">`, à la place du 5267-f73dcd000abcc353. Après avoir soumis l'URL, on rafraichit la page et on obtient alors une alerte pop-up XSS.

API-Only XSS	XSS	***	Réaliser une attaque XSS persistante avec `<iframe src="javascript:alert(`xss`)">` sans utiliser l'application front-end.	Il faut se logger à l'application avec n'importe quel utilisateur et utiliser Burp. On copie l'en-tête « Authorization » de n'importe quelle requête venant du navigateur. Puis on soumet une requête POST vers http://localhost:3000/api/Products avec : `{"name": "XSS", "description": "<iframe src=\"javascript:alert(xss)\">", "price": 47.11}` en corps, *application/json* en Content-Type *Bearer ?* comme en-tête Authorization, remplaçant le ? avec le token copié du navigateur.

XXE Data Access	XXE	***	Une attaque XXE est un type d'attaque contre une application qui analyse l'entrée XML. Cette attaque se produit lorsque l'entrée XML contenant une référence à une entité externe est traitée par un analyseur XML faiblement configuré. Cette attaque peut conduire à la divulgation de données confidentielles, à un déni de service, à une falsification de requête côté serveur, à un balayage de port du point de vue de la machine où se trouve l'analyseur, et à d'autres impacts sur le système. Ici le but est de récupérer le contenu du fichier C:\Windows\system.ini ou /etc/passwd sur le serveur.	Il faut ici faire une injection XML pour extraire des données à distance, avec un code comme suit :
				`<?xml version="1.0" encoding="ISO-8859-1"?>`
				`<!DOCTYPE foo [`
				`<!ELEMENT foo ANY >`
				`<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>`
				`<foo>&xxe;</foo>`
				On peut le faire dans le formulaire « complaint » en insérant un fichier d'injection pour voir le résultat avec l'aide de Burp.