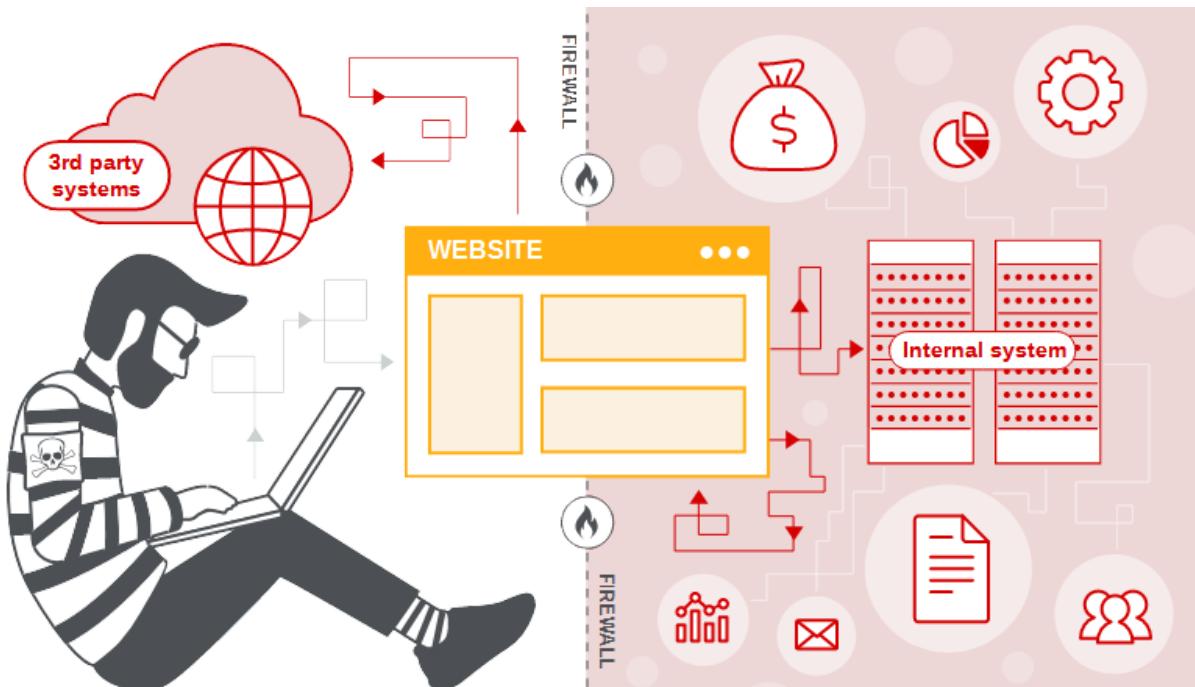




Université Cheikh Anta Diop de Dakar (UCAD)  
Ecole Supérieure Polytechnique (ESP)  
Département Génie Informatique  
**Module:** Cryptographie

## Sujet: Server Side Request Forgery (SSRF)



**Professeur:**  
Gervais Mendy

**Groupe 10:**  
El Hadj T. Keita THIAM  
Serigne Saliou LO  
Rokhaya SEMBENE  
Saliou BOP

# **Table Des Matières**

I. Définition	3
II. Matching SSRF	5
III. CVE - CWE	6
IV. Exemples De Scénario	7
V. Cas Pratique	8

# I. Définition

La contrefaçon de requête côté serveur (également connue sous le nom de **SSRF**) est une vulnérabilité de sécurité Web qui permet à un attaquant d'inciter l'application côté serveur à effectuer des requêtes vers un emplacement non souhaité.

Dans une attaque **SSRF** typique, l'attaquant peut amener le serveur à établir une connexion avec des services internes uniquement au sein de l'infrastructure de l'organisation. Dans d'autres cas, ils peuvent être en mesure de forcer le serveur à se connecter à des systèmes externes arbitraires, ce qui risque de divulguer des données sensibles telles que les informations d'identification d'autorisation.

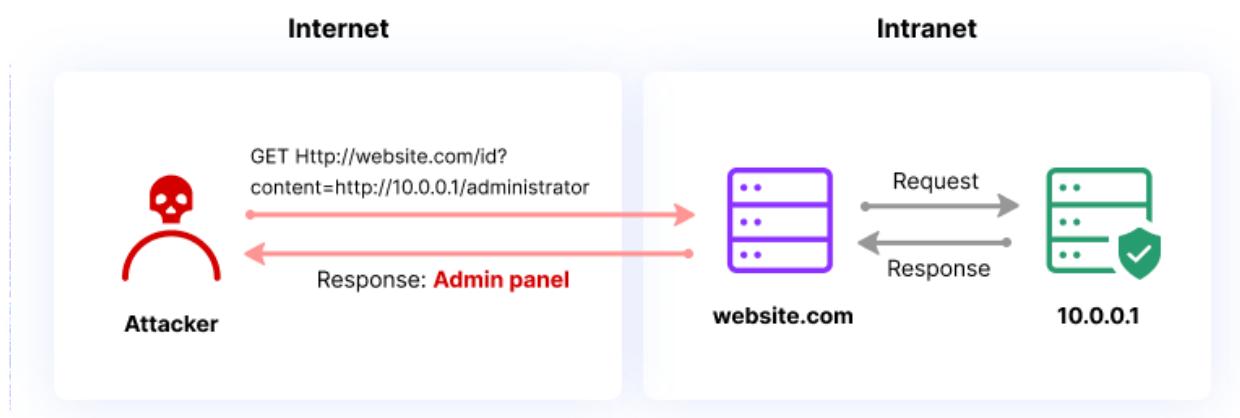


Figure 1: Attaque SSRF

Comme les applications Web modernes offrent aux utilisateurs finaux des fonctionnalités pratiques, la récupération d'une URL devient un scénario courant. En conséquence, l'incidence de la **SSRF** augmente. De plus, la sévérité de **SSRF** est de plus en plus élevée en raison des services cloud et de la complexité des architectures.

## Analyse des Facteurs

CWE associés communément appelé Common Weakness Enumeration sont les faiblesses logicielles ou matérielles liées à la SSRF et développées par la communauté CWETM.

**Taux d'incidence maximum** est le rapport du nombre maximum de nouveaux cas de SSRF détectés pendant une période donnée au nombre de cas total de SSRF découverts.

**Taux d'incidence moyen** est le rapport du nombre moyen de nouveaux cas de SSRF détectés pendant une période donnée au nombre de cas total de SSRF découverts.

**Exploitation pondérée moyenne** est la moyenne du nombre de cas de la SSRF chacun multiplié par le nombre d'exploitations de ce cas le tout sur le nombre total d'exploitations.

**Impact pondéré moyen** est la moyenne du nombre de types d'impacts de la SSRF chaque type multiplié par le nombre total d'impacts qu'il a eu.

**Couverture maximum** est le nombre maximum de systèmes informatiques touchés pas la SSRF sur les échantillons prélevées sur le nombre total de sys.

**Couverture moyenne** est le est le nombre moyen systèmes informatiques touchés pas la SSRF sur le nombre total.

**Nombre total d'occurrences** est le nombre total de répétitions de la SSRF dans un système informatique.

**Nombre total de CVEs** est le nombre total de faiblesses logicielles ou matérielles développées par la communauté CWETM.

CWE	Taux d'Incidence Maximum	Taux d'Incidence Moyenne	Exploitation pondérée moyenne	Impact pondéré moyen	Couverture maximum	Couverture moyenne	Nombre total d'occurrences	Nombre total de CVEs
1	2.72%	2.72%	8.28	6.72	67.72%	67.72%	9503	385

Tableau 1: Facteurs

## II. Matching SSRF

La correspondance entre le SSRF et la cryptographie en générale est associée à la transgression des critères sécurité suivants:

### **Confidentialité:**

La confidentialité fait référence aux efforts d'une organisation pour garder ses données privées ou secrètes, dans une politique de confidentialité par exemple. En pratique, il s'agit de contrôler l'accès aux données pour empêcher leur divulgation non autorisée. En règle générale, cela implique de s'assurer que seuls ceux qui sont autorisés ont accès à des actifs spécifiques et que ceux qui ne le sont pas sont activement empêchés d'obtenir l'accès

### **Authentifications:**

L'authentification empêche l'usurpation d'identité et oblige les utilisateurs à confirmer leur identité avant d'être autorisés à accéder aux systèmes et aux ressources. Cela inclut les noms d'utilisateur, les mots de passe, les e-mails, la biométrie et autres.

### **Intégrité:**

L'intégrité consiste à s'assurer que les données n'ont pas été falsifiées et qu'elles sont donc correctes, authentiques et fiables. Les clients du commerce électronique, par exemple, s'attendent à ce que les informations sur les produits et les prix soient exacts, et que la quantité, les prix, la disponibilité et d'autres informations ne soient pas modifiés après avoir passé une commande. Les clients des banques doivent pouvoir être sûrs que leurs informations bancaires et les soldes de leurs comptes n'ont pas été falsifiés.

### **Problèmes:**

Le problème de confidentialité est matérialisé par le fait que l'attaquant puisse avoir accès aux informations tels que les ports et adresses IP afin d'envoyer des requêtes sans détenir le droit.

Le problème d'intégrité des données fait surface lorsque l'application web récupère une ressource distante interceptée par l'attaquant sans s'en rendre compte car ici le serveur web ne peut guère distinguer une fausse requête d'une vraie.

## III. CVE - CWE

### CWE-918 SSRF

Dans ce CWE précis le serveur Web reçoit une URL ou une demande similaire d'un composant en amont et récupère le contenu de cette URL, mais il ne s'assure pas suffisamment que la demande est envoyée à la destination prévue.

Tout d'abord il y'a non lieu d'authentification car le serveur n'est pas sûr que l'information soit arrivée à destination et un non lieu de confidentialité car le serveur Web n'est guère sûr que l'information n'est accessible qu'à ceux dont l'accès est autorisé. Pas en phase avec le chiffrement classique car le contenu du message est retrouvé pas des personnes auxquelles le message n'est point adressé.

### CVE 2002-1484

Le serveur DB4Web, lorsqu'il est configuré pour utiliser des messages de débogage verbeux, permet aux attaquants distants d'utiliser DB4Web en tant que proxy et de tenter des connexions TCP à d'autres systèmes (analyse de port) via une demande d'URL qui spécifie l'adresse IP cible et le port, ce qui produit un statut de connexion dans le message d'erreur résultant.

Dans ce cas nous avons un problème d'authentification car l'intrus (l'attaquant) se fait passer pour un système hôte du serveur DB4web et également un problème de confidentialité car l'attaquant a accès aux informations tels que les ports et adresses IP ainsi qu'un problème d'intégrité des données car les tentatives de connexion de l'attaquant ont réussi donc le service est incapable de discerner un faux message d'un message légitime.

### CVE-2021-26855

Il s'agit d'une vulnérabilité SSRF dans Microsoft Exchange Server. Un attaquant distant non authentifié pourrait exploiter cette faille en envoyant une demande HTTP spécialement conçue à un serveur Exchange vulnérable. Afin d'exploiter cette faille, Microsoft dit que le serveur Exchange vulnérable devrait être en mesure d'accepter des connexions non fiables sur le port 443. L'exploitation réussie de cette faille permettrait à l'attaquant de s'authentifier auprès du serveur Exchange.

Nous signalons dans ce cas un problème de non authentification de l'attaquant, un problème d'intégrité de données car le serveur ne peut discerner une connexion fiable d'une autre non fiable.

## IV. Exemples De Scénarios

Nous allons aborder plusieurs scénarios pour mieux comprendre l'attaque **SSRF**

### **Scénario 1:** Serveurs internes d'analyse de port

Si l'architecture du réseau n'est pas segmentée, les attaquants peuvent cartographier les réseaux internes et déterminer si les ports sont ouverts ou fermés sur les serveurs internes à partir des résultats de connexion ou du temps écoulé pour se connecter ou rejeter les connexions de charge utile SSRF.

### **Scénario 2:** Exposition des données sensibles

Les attaquants peuvent accéder aux fichiers locaux ou aux services internes pour obtenir des informations sensibles telles que <file:///etc/passwd> et <http://localhost:28017/>

### **Scénario 3:** Accéder au stockage des métadonnées des services cloud

La plupart des fournisseurs de cloud disposent d'un stockage de métadonnées tel que <http://169.254.169.254/>. Un attaquant peut lire les métadonnées pour obtenir des informations sensibles

### **Scénario 4:** Compromettre les services internes

L'attaquant peut abuser des services internes pour mener d'autres attaques telles que l'exécution de code à distance (RCE) ou le déni de service (DoS).

# V. Cas Pratique

Nous allons présenter ici une application que nous avons développée pour explorer quelques scénarios de vulnérabilités SSRF.

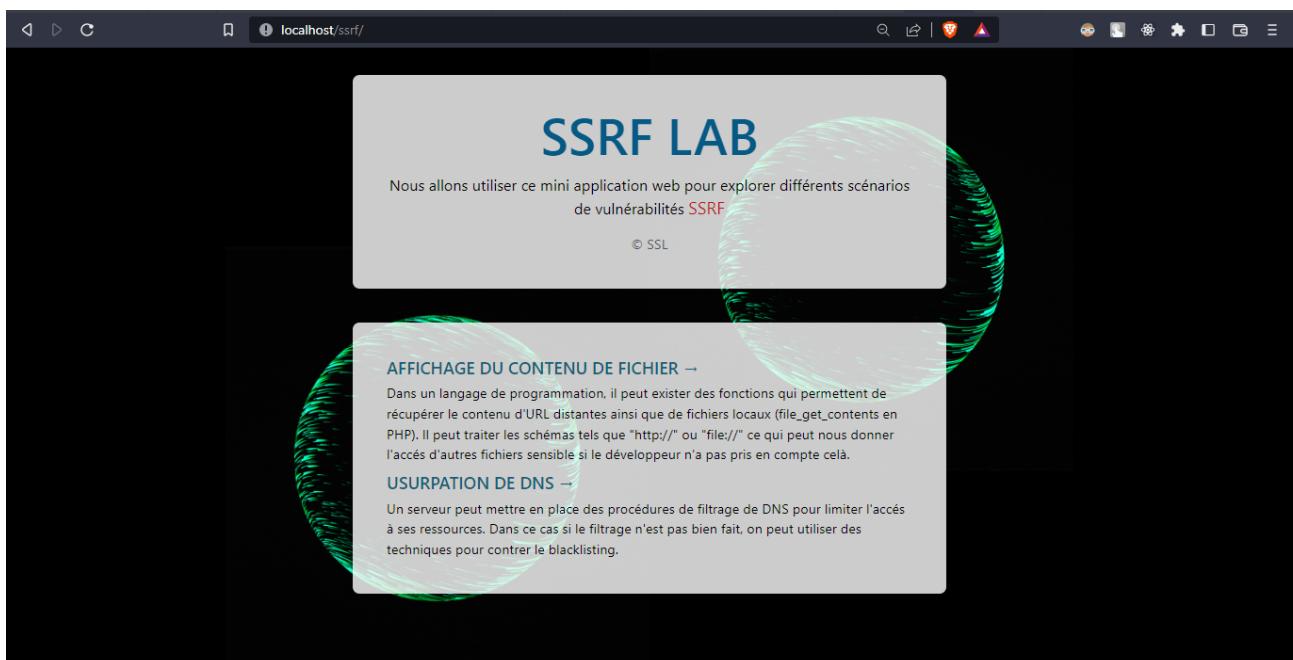


Figure 2: Page d'accueil de l'application

Lien vers l'application web: <https://github.com/saliou33/ssrf>

L'application présente pour l'instant 2 scénarios:

## 1. Affichage du contenu de Fichier

Dans un langage de programmation, il peut exister des fonctions qui permettent de récupérer le contenu d'URL distantes ainsi que de fichiers locaux (**file\_get\_contents** en PHP). Il peut traiter les schémas tels que "**http://**" ou "**file://**" ce qui peut nous donner l'accès d'autres fichiers sensibles si le développeur n'a pas pris en compte cela.

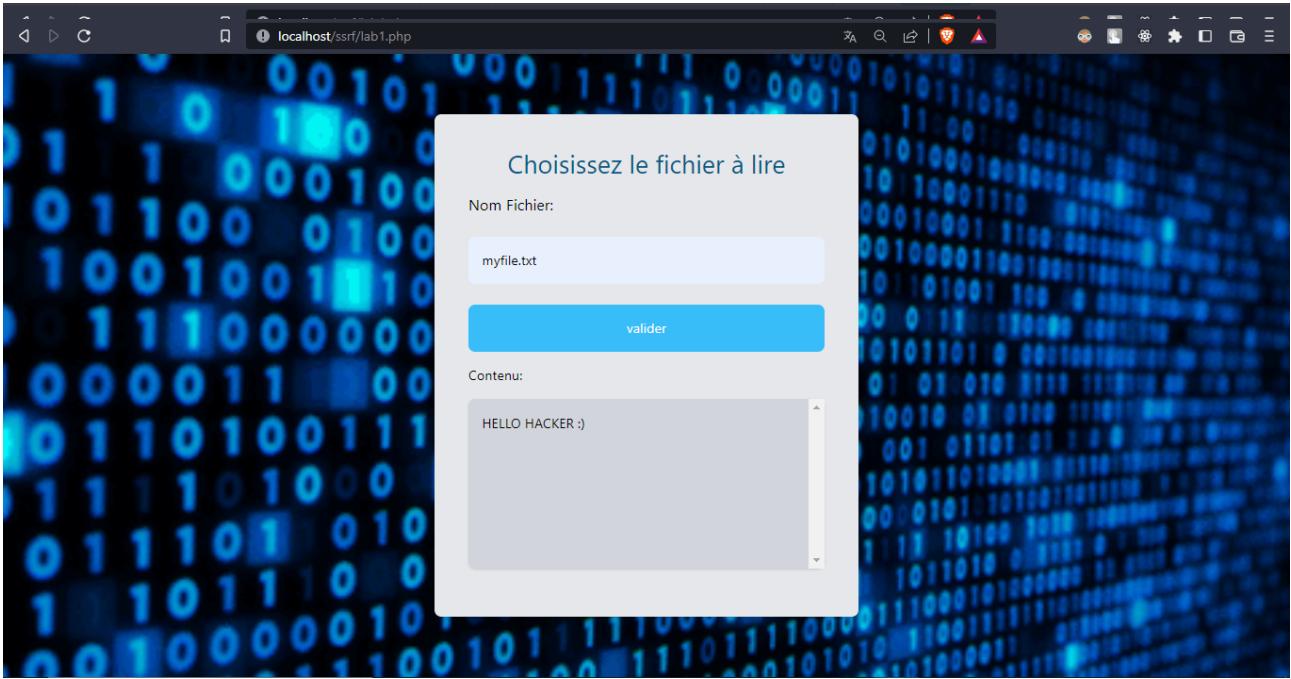


Figure 3: Page lab1.php

Dans le répertoire du projet, il y a un fichier **myfile.txt**; son contenu est en premier lieu afficher pour démontrer la fonction **file\_get\_contents**. Maintenant vu que le 'développeur' est insouciant, il n'a pas pris le temps de préfixer le nom du fichier par une chaîne représentant le répertoire parent des fichiers accessibles. Nous pouvons utiliser celà pour accéder à des informations sensibles illustrées par la figure suivante.

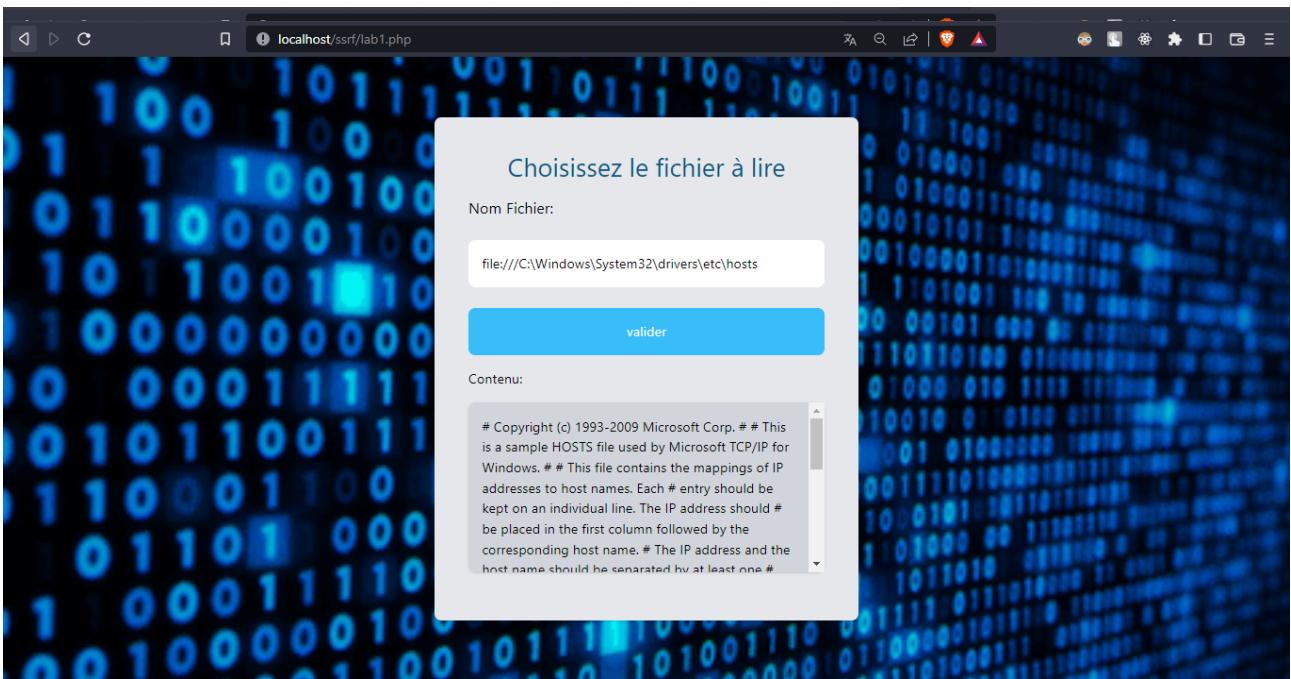


Figure 4: Exploitation de la faille

En lui donnant comme input `file:///C:/Windows/System32/drivers/etc/hosts` on arrive à accéder à son contenu; ainsi récupérer des informations sensibles sur les hôtes.

## 2. Usurpation de DNS

Un serveur peut mettre en place des procédures de filtrage de DNS pour limiter l'accès à ses ressources. Dans ce cas, si le filtrage n'est pas bien fait, on peut utiliser des techniques pour contrer le blacklisting.

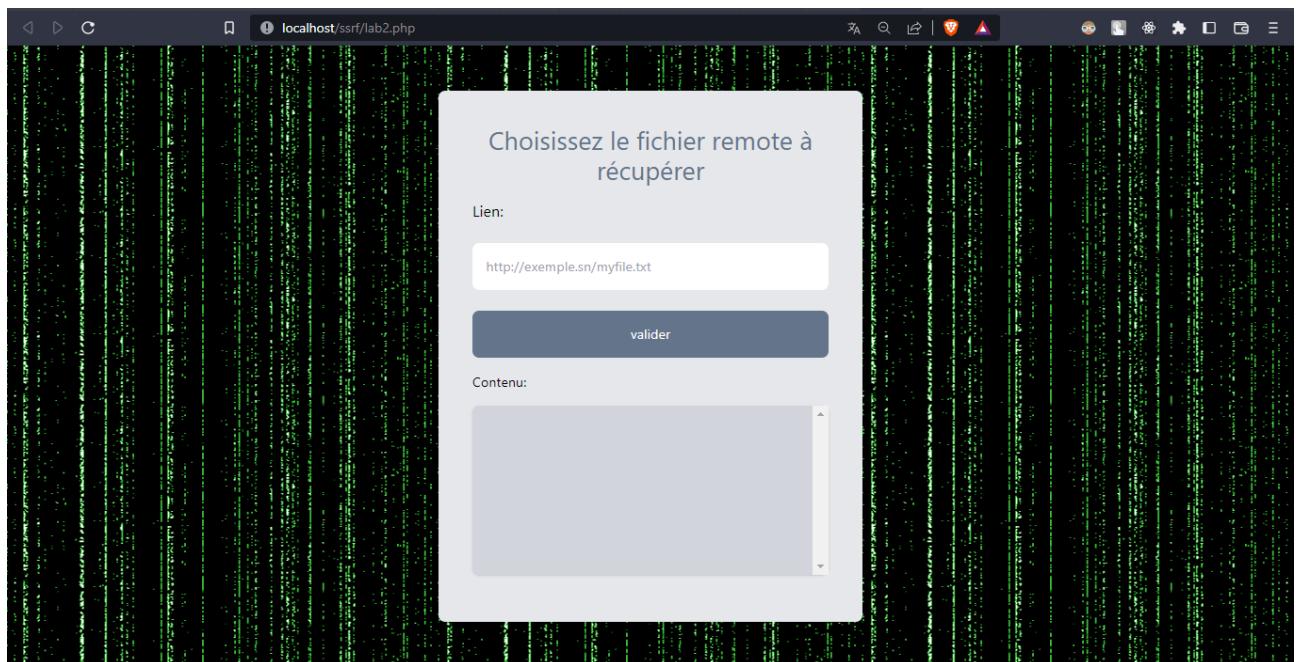


Figure 5: Page lab2.php

Dans ce scénario, l'application permet d'afficher le contenu de fichier distant. Notre objectif est de gagner l'accès sur le fichier **myfile.txt** du serveur. L'application effectue un filtrage d'url pour ne pas donner d'accès à ses fichiers internes.

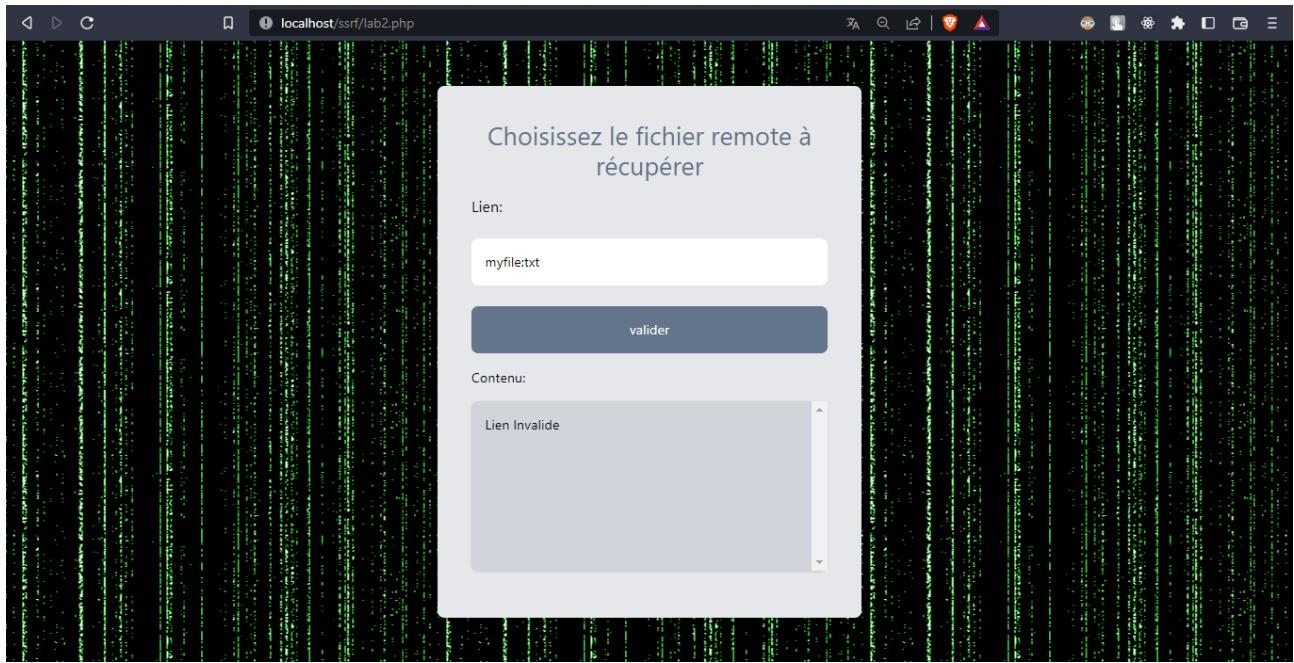


Figure 6: Erreur accès myfile.txt

L'application affiche un message d'erreur lorsque lien est invalide vu qu'il doit être préfixer par **http** ou **https**.

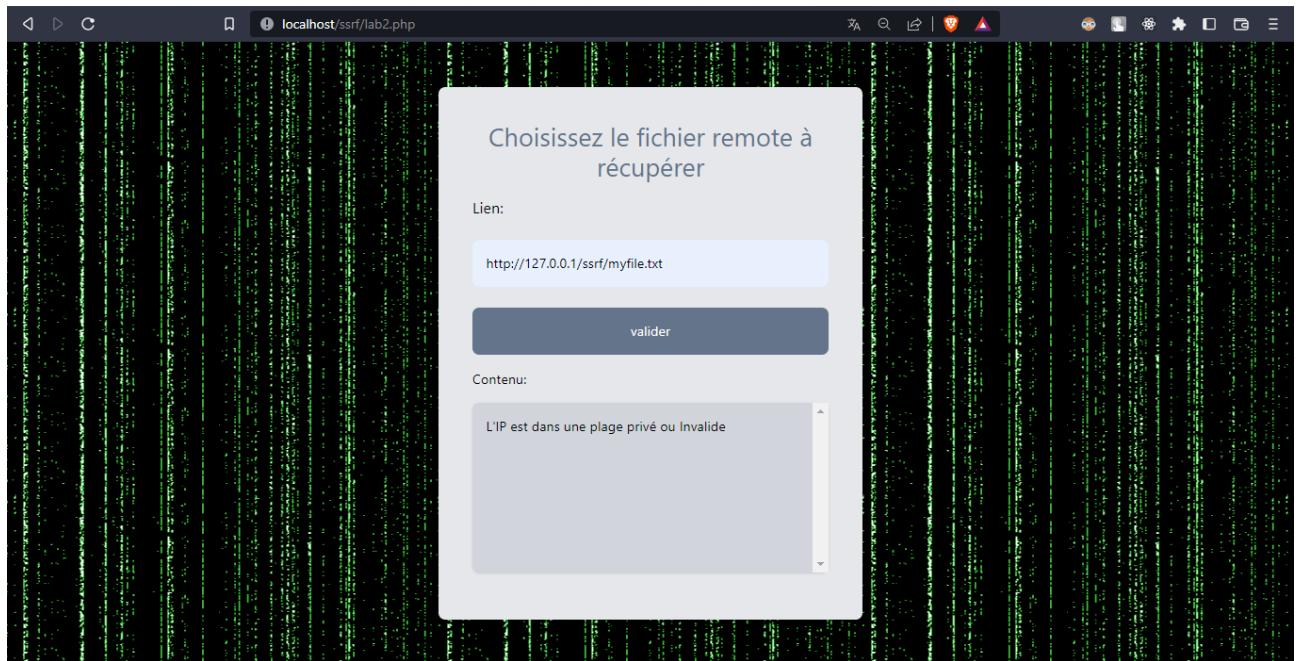


Figure 7: Erreur Ip

En essayant de passer avec l'adresse IP, le hacker n'arrive pas à voir le contenu du fichier.

Ainsi pour contrecarrer ce dispositif de filtrage, nous pouvons attacher l'adresse IP **127.0.0.1** avec un autre nom de domaine (**dnsusurp.sn**) pour bypasser le filtrage de l'url ainsi finalement accéder au contenu du fichier **myfile.txt**.

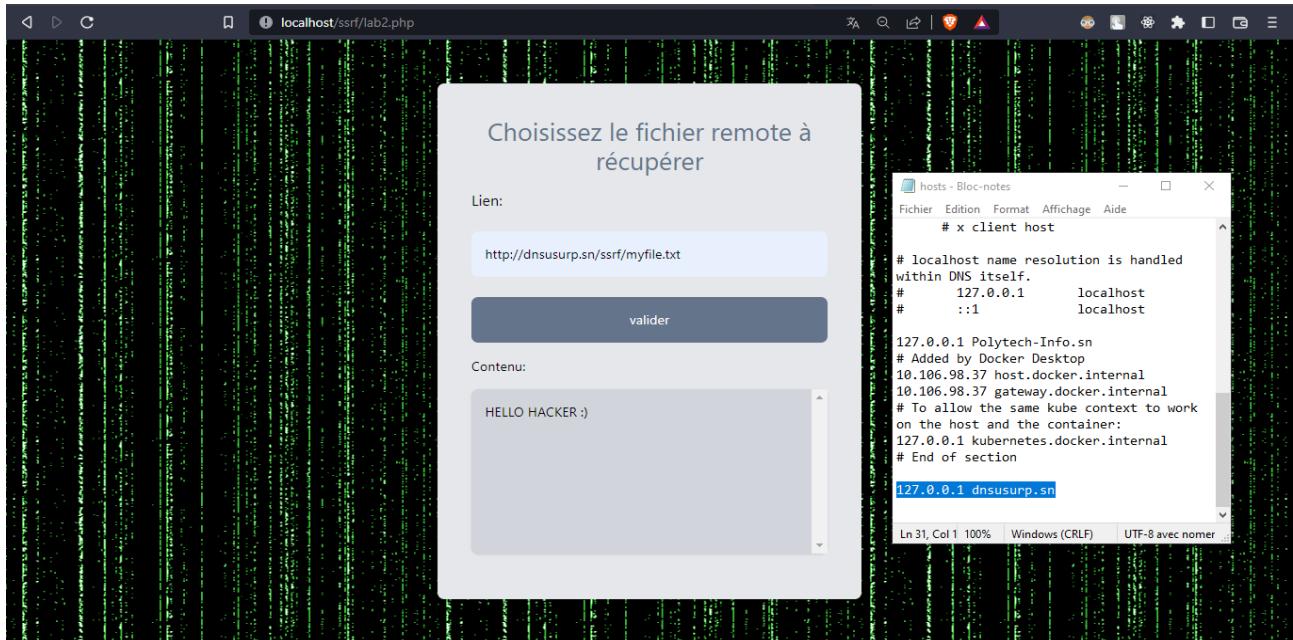


Figure 8: Usurpation de Domain Name