

# Pentest Vulnerability Report : Beta

## Table des matières

<i>Cible</i> .....	3
<i>Attaquant</i> .....	3
<i>Titre de la vulnérabilité</i> .....	3
<i>Description de la vulnérabilité</i> .....	3
<i>Éléments affectés</i> .....	3
<i>Préalables</i> .....	3
<i>Mise en place</i> .....	3
<i>Proof of concept</i> .....	4
<i>Impact</i> .....	6
<i>Mitigation</i> .....	6

## Cible

Le domaine local « rs.io »

## Attaquant

Les Rogue Sentinels

## Titre de la vulnérabilité

Reverse engineering

## Description de la vulnérabilité

Cette attaque consiste à remplacer une fonction qui devait être exécutée par une autre.

## Éléments affectés

Le fichier « tests » trouvé grâce à un accès anonyme au service VSFTP.

## Préalables

- ➔ Un toolkit qui nous est donné.
- ➔ Réseau de machines. (rs.io)

## Mise en place

Ce proof of concept est conçu sur une machine virtuelle Kali Linux 64 bits.

Notre cible est donnée, il s'agit du domaine local « rs.io ».

Nous installons le toolkit fourni sur un Kali Linux 64-bit avec cette ligne de commande :

- ➔ `wget https://raw.githubusercontent.com/RogueSentinels/hacker-toolkit/main/attack.sh`  
`&& chmod +x attack.sh`

Nous préparons notre station de travail avec :

➔ `sudo ./attack.sh workstation-setup`

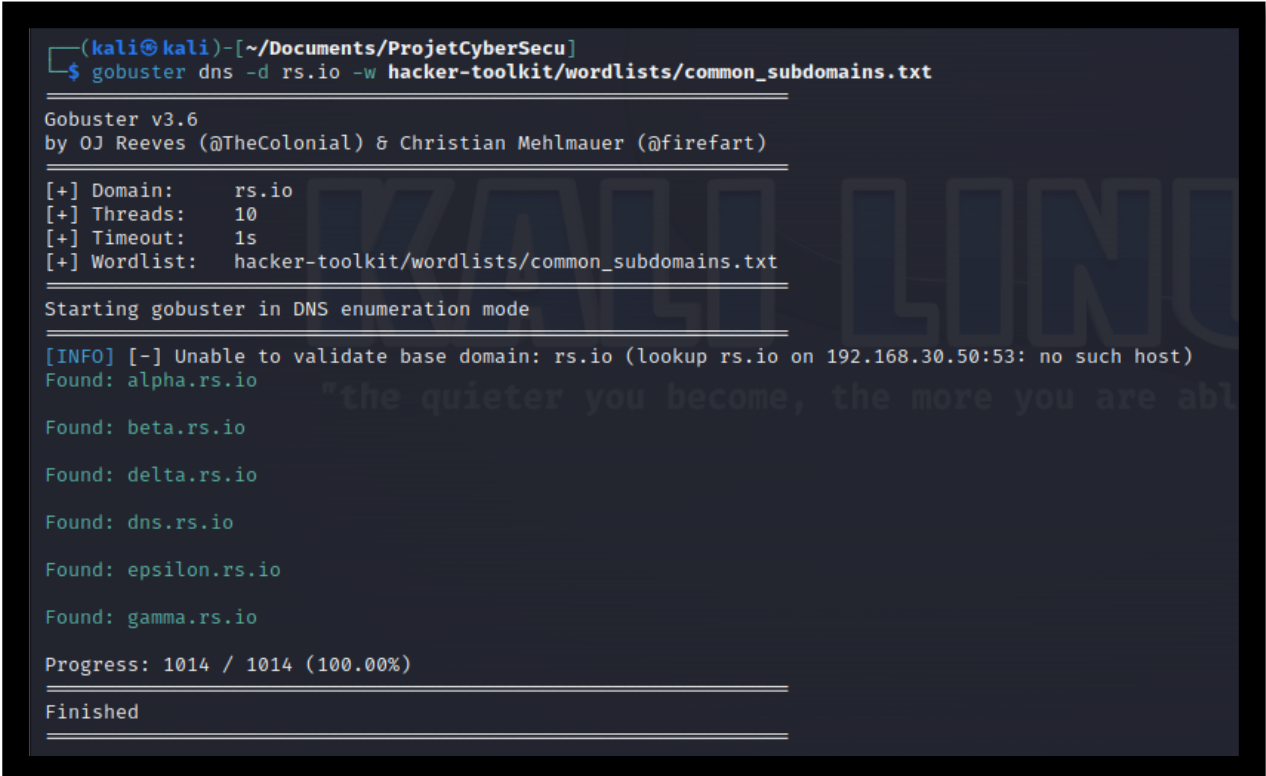
Et nous lançons l'attaque avec :

➔ `sudo ./attack.sh up`

## Proof of concept

1) Chercher les sous domaines avec gobuster :

➔ `gobuster dns -d rs.io -w hacker-toolkit/wordlists/common_subdomains.txt`



```
(kali㉿kali)-[~/Documents/ProjetCyberSecu]
└─$ gobuster dns -d rs.io -w hacker-toolkit/wordlists/common_subdomains.txt

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Domain:      rs.io
[+] Threads:     10
[+] Timeout:     1s
[+] Wordlist:     hacker-toolkit/wordlists/common_subdomains.txt

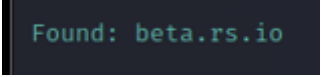
Starting gobuster in DNS enumeration mode

[INFO] [-] Unable to validate base domain: rs.io (lookup rs.io on 192.168.30.50:53: no such host)
Found: alpha.rs.io
Found: beta.rs.io
Found: delta.rs.io
Found: dns.rs.io
Found: epsilon.rs.io
Found: gamma.rs.io

Progress: 1014 / 1014 (100.00%)

Finished
```

➔ 2) Trouver le sous domaine `beta.rs.io`



```
Found: beta.rs.io
```

3) Scan de port avec nmap sur le sous-domaine :

➔ `nmap -p- beta.rs.io`

4) On trouve le port 21 ouvert.

5) On se connecte sur le port 21 avec un client ftp :

➔ `ftp beta.rs.io`

5) On trouve un fichier « tests » qui est un programme en C.

6) On récupère le fichier « tests » :

➔ `get tests`

7) On ouvre le fichier « tests » avec gdb :

➔ `gdb tests`

8) On regarde quelles fonctions sont présentes dans le programme :

➔ `info functions`

9) On trouve la fonction `main` et on regarde son code assembleur :

➔ `disassemble main`

10) On voit que le programme exécute la fonction « no ».

11) On regarde le code assembleur de la fonction « no » :

➔ `disassemble no`

12) On exécute le programme avec gdb :

➔ run

13) Cette fonction « no » ne nous donne pas le flag, donc on va essayer avec la fonction « test », pour cela on va modifier le code assembleur de la fonction « main » pour qu'elle exécute la fonction « test » au lieu de la fonction « no ».

14) On met un breakpoint sur la fonction « no » :

➔ break no

15) On modifie \$rip pour qu'il pointe sur la fonction « test » avec la troisième adresse (skip le setup de la fonction de return) trouvée lors de l'exécution de la commande « info functions » :

➔ set \$rip = 0x00005555555513d

16) On exécute le programme :

➔ continue

17) On trouve le flag dans la sortie du programme :

➔ Flag = \${FLAG\_Reverso}

## Impact

En changeant la méthode qui devait être exécutée à la base, nous modifions le comportement du programme.

## Mitigation

Il faut enlever l'accès anonyme au service VSFTP.