



**Université  
de Limoges**

**UNIVERSITY OF LIMOGES**

Faculty of Science and Technology

**Master 2 CRYPTIS**

**Sécurité de l'Information et Cryptologie  
(CRYPTIS)**

*Parcours Informatique*

Administration et sécurité des réseaux et systèmes - Semestre I

---

**IPsec**

---

**SALAME Joe**

*Enseignants*

**M. CONCHON Emmanuel**

27 février 2024

# Chapitre 1

## Static

### 1.1 Mode Tunnel

Dans le mode tunnel, le paquet IP d'origine est chiffré et encapsulé dans un nouveau paquet IP. Le mode tunnel est couramment utilisé pour créer des connexions VPN. Ce mode ajoute une couche de sécurité supplémentaire en cachant l'adresse IP source et de destination d'origine, ce qui rend plus difficile la collecte d'informations sur le réseau.

Premièrement j'ai généré les identifiants SPI et RID en utilisant Openssl . Les clés générées (k1, k2, k3, k4) sont des chaînes hexadécimales de 32 octets. Les SPI et les RID sont des chaînes hexadécimales de 4 octets, tous préfixés par "0x" pour indiquer leur format hexadécimal.

Les deux premières commandes ajoutent des états IPsec dans le namespace h1. Ces états spécifient les paramètres de sécurité pour le chiffrement et l'authentification des paquets IP envoyés de h1 à h2 et vice-versa. Les valeurs de SPI et de RID sont utilisées pour identifier ces états. Ensuite j'ai ajouté des politiques IPsec dans h1. Ces politiques définissent quelles sont les communications qui doivent être sécurisées (du point de vue de h1) et quelles sont les règles à appliquer pour sécuriser ces communications. Elles spécifient également les paramètres de tunnel pour la sécurité.

```
1 ip netns exec h1 ip xfrm state add src 10.10.10.1 dst 10.0.0.1 proto esp spi
   $SPI1 reqid $RID1 mode tunnel auth sha256 $k1 enc aes $k2
2 ip netns exec h1 ip xfrm state add src 10.0.0.1 dst 10.10.10.1 proto esp spi
   $SPI2 reqid $RID2 mode tunnel auth sha256 $k3 enc aes $k4
3
4 ip netns exec h1 ip xfrm policy add src 10.0.0.1 dst 10.10.10.1 dir in tmpl src
   10.0.0.1 dst 10.10.10.1 proto esp reqid $RID2 mode tunnel
5 ip netns exec h1 ip xfrm policy add src 10.10.10.1 dst 10.0.0.1 dir out tmpl src
   10.10.10.1 dst 10.0.0.1 proto esp reqid $RID1 mode tunnel
```

J'ai fait la même chose pour la configuration du namespace h2.

Dans la capture d'écran ci-dessous, j'ai exécuté une commande socat sur les deux hôtes h1 et h2 sans utiliser IPsec. Comme nous pouvons le voir, la communication n'a pas été chiffrée et le message est en claire dans le segment TCP.

```

vlad@vlad:~$ sudo ip netns exec h1 socat - tcp-listen:6789
HELLOO

vlad@vlad:~$ sudo ip netns exec r1 tcpdump -lnvX "ip"
tcpdump: listening on r1-eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
14:55:26.784788 IP (tos 0x0, ttl 62, id 12327, offset 0, flags [DF], proto TCP (6), length 60)
10.0.0.1.54392 > 10.10.10.1.6789: Flags [P-], cksum 0xb376 (correct), seq 3481230329, ack
584493172, win 502, options [nop,nop,TS val 2355109604 ecr 4243964469], length 8
0x0000: 4500 003c 3027 4000 3e06 ee89 0a00 0001 E..<0'@>.....
0x0010: 0a0a 0a01 d478 1a85 c7ff 5bf1 22d6 a874 ....X...[...].t
0x0020: 8010 01f6 b376 0000 0101 080a 8c00 1ee4 ....V.....
0x0030: fcfc c235 4845 6c0c 4f4f 4f0a ...SHELL000.
14:55:26.784840 IP (tos 0x0, ttl 64, id 38286, offset 0, flags [DF], proto TCP (6), length 52)
10.10.10.1.6789 > 10.0.0.1.54392: Flags [..], cksum 0xda8b (correct), seq 1, ack 8, win 509, option
s [nop,nop,TS val 4243975724 ecr 2355109604], length 0
0x0000: 4500 0034 958e 4000 4006 892a 0a0a 0a01 E..4..@>.....
0x0010: 0a00 0001 1a85 d478 22d6 a874 c7ff 5bf9 .....X'..t.[.
0x0020: 8010 01f6 da8b 0000 0101 080a fcfc ee2c ....V.....
0x0030: 8c00 1ee4 ....

```

```

vlad@vlad:~$ sudo ip netns exec h2 socat tcp:10.10.10.1:6789 -
HELLOO

vlad@vlad:~$ sudo ip netns exec r2 tcpdump -lnvX "ip"
tcpdump: listening on r2-eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
14:55:26.784738 IP (tos 0x0, ttl 64, id 12327, offset 0, flags [DF], proto TCP (6), length 60)
10.0.0.1.54392 > 10.10.10.1.6789: Flags [P-], cksum 0xb376 (correct), seq 3481230329, ack
584493172, win 502, options [nop,nop,TS val 2355109604 ecr 4243964469], length 8
0x0000: 4500 003c 3027 4000 4006 ec89 0a00 0001 E..<0'@>.....
0x0010: 0a0a 0a01 d478 1a85 c7ff 5bf1 22d6 a874 ....X...[...].t
0x0020: 8010 01f6 b376 0000 0101 080a 8c00 1ee4 ....V.....
0x0030: fcfc c235 4845 6c0c 4f4f 4f0a ...SHELL000.
14:55:26.784863 IP (tos 0x0, ttl 62, id 38286, offset 0, flags [DF], proto TCP (6), length 52)
10.10.10.1.6789 > 10.0.0.1.54392: Flags [..], cksum 0xda8b (correct), seq 1, ack 8, win 509, option
s [nop,nop,TS val 4243975724 ecr 2355109604], length 0
0x0000: 4500 0034 958e 4000 3e06 892a 0a0a 0a01 E..4..@>.....
0x0010: 0a00 0001 1a85 d478 22d6 a874 c7ff 5bf9 .....X'..t.[.
0x0020: 8010 01f6 da8b 0000 0101 080a fcfc ee2c ....V.....
0x0030: 8c00 1ee4 ....

```

## Sans IPsec

Comme vous remarquez ci-dessous, après l'implémentation d'IPsec nous ne pouvons plus lire le message transmis à l'aide de socat vu qu'il est chiffré.

```

vlad@vlad:~$ sudo ip netns exec h1 socat - tcp-listen:6789
HELLOO

vlad@vlad:~$ sudo ip netns exec r1 tcpdump -lnvX "ip"
tcpdump: listening on r1-eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
14:18:57.850278 IP (tos 0x0, ttl 62, id 0, offset 0, flags [DF], proto ESP (50), length 120)
10.0.0.1 > 10.10.10.1: ESP(spi=0x2f7fa004,seq=0x7), length 100
0x0000: 4500 0078 0000 4000 3e32 1c49 0a00 0001 E..X..@>2.I....
0x0010: 0a0a 0a01 2ff8 4fae 0000 0009 6d48 25b1 .../..O.....MHK.
0x0020: c0cf 3763 8278 b614 014d 84fd 9504 e76f ...7c.z...A.....0
0x0030: 97c5 1f24 01d6 1049 30d5 d7b2 fcc7 17af ...S...I0.....
0x0040: 89a0 8c80 7711 0fc7 2cc9 16ec c4a8 01ef ...W.....
0x0050: 15b3 866f 01eb 3c44 6e52 2c93 e695 b4dd ...o..<dnR.....
0x0060: 1fd0 e15b 439c 2b92 7c87 cc75 4913 af5a ...[C+..].UI..Z
0x0070: 6e63 5f1b 7296 ffae fc.....
14:18:57.850423 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ESP (50), length 120)
10.10.10.1 > 10.0.0.1: ESP(spi=0x2f7fa004,seq=0x7), length 100
0x0000: 4500 0078 0000 4000 4032 1c49 0a0a 0a01 E..X..@>2.I....
0x0010: 0a00 0001 e90f a004 0000 0007 4d25 d5cf .....MK.....
0x0020: d07f 3025 59c2 5a89 dfb1 3fac f537 1a08 ...0NY.Z...7..7..
0x0030: c502 07aa 0ac6 e600 bb4f ed3c 0e41 7eab ..g....O.<.A...
0x0040: 0b46 ec09 accf cdca cf50 8c29 c4da 0310 kf.....V.....
0x0050: d1c5 c133 c158 16ee aa88 d1ce d50d aa01 ...3.X.....
0x0060: 35e2 bc89 1e72 b150 320f c489 c9b6 5197 5....r.P2....Q.
0x0070: 28e9 4f64 824b 6e9f (.Od.Kn.

```

```

vlad@vlad:~$ sudo ip netns exec h2 socat tcp:10.10.10.1:6789 -
Helloo

vlad@vlad:~$ sudo ip netns exec r2 tcpdump -lnvX "ip"
tcpdump: listening on r2-eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
14:18:57.850201 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ESP (50), length 120)
10.0.0.1 > 10.10.10.1: ESP(spi=0x2f7fa004,seq=0x7), length 100
0x0000: 4500 0078 0000 4000 4032 1c49 0a00 0001 E..X..@>2.I....
0x0010: 0a0a 0a01 2ff8 4fae 0000 0009 6d48 25b1 .../..O.....MHK.
0x0020: c0cf 3763 8278 b614 014d 84fd 9504 e76f ...7c.z...A.....0
0x0030: 97c5 1f24 01d6 1049 30d5 d7b2 fcc7 17af ...S...I0.....
0x0040: 89a0 8c80 7711 0fc7 2cc9 16ec c4a8 01ef ...W.....
0x0050: 15b3 866f 01eb 3c44 6e52 2c93 e695 b4dd ...o..<dnR.....
0x0060: 1fd0 e15b 439c 2b92 7c87 cc75 4913 af5a ...[C+..].UI..Z
0x0070: 6e63 5f1b 7296 ffae fc.....
14:18:57.850490 IP (tos 0x0, ttl 62, id 0, offset 0, flags [DF], proto ESP (50), length 120)
10.10.10.1 > 10.0.0.1: ESP(spi=0x2f7fa004,seq=0x7), length 100
0x0000: 4500 0078 0000 4000 3e32 1c49 0a0a 0a01 E..X..@>2.I....
0x0010: 0a00 0001 e90f a004 0000 0007 4d25 d5cf .....MK.....
0x0020: d07f 3025 59c2 5a89 dfb1 3fac f537 1a08 ...0NY.Z...7..7..
0x0030: c502 07aa 0ac6 e600 bb4f ed3c 0e41 7eab ..g....O.<.A...
0x0040: 0b46 ec09 accf cdca cf50 8c29 c4da 0310 kf.....V.....
0x0050: d1c5 c133 c158 16ee aa88 d1ce d50d aa01 ...3.X.....
0x0060: 35e2 bc89 1e72 b150 320f c489 c9b6 5197 5....r.P2....Q.
0x0070: 28e9 4f64 824b 6e9f (.Od.Kn.

```

## IPsec Mode Tunnel

## 1.2 Mode Transport

Dans le mode transport, seuls les paquets de données (payload) sont chiffrés ou signés. L'en-tête IP d'origine est conservé, bien que certaines informations puissent être ajoutées. Le mode transport est souvent utilisé pour sécuriser la communication point à point entre deux hôtes. Le mode transport est plus efficace en termes de traitement, car il ne chiffre que les données elles-mêmes, pas les en-têtes IP.

Pour changer en mode transport nous n'avons qu'à changer le mode dans la configuration

de la policy et du state.

```

vlad@vlad:~$ sudo netcat -l -p 4444
Hello
Hello

vlad@vlad:~$ sudo netcat -l -p 4444
Hello
Hello

IPsec Mode Transport

0x0030: 8fde 9eeb 5a90 76dc fd60 19ad 1171 eb98 .m..Z.V.....q..
0x0040: f3be e339 893e 6273 4228 5539 5f44 62fc ...9..>ba8(U9_Db.
0x0050: c355 e015 65d1 6a4d 47ab 518c ab1c d872 ..u..e.jmG.Q....r
0x0060: f0c2 16bf e500 efc4 .....
14:51:40.271817 IP (tos 0x0, ttl 62, id 40890, offset 0, flags [DF], proto ESP (50), length 104)
10.0.0.1 > 10.10.10.1: ESP(spi=0xb16474e,seq=0x3), length 84
0x0000: 4500 0068 9fba 4000 3e32 7e9e 0a00 0001 E..h..0..>2.....
0x0010: 0a0a 0a01 a6f7 a457 0000 0004 03a6 7412 .....W.....t.
0x0020: 9ed7 8925 7226 80a0 33e0 b33f 7070 4030 ...NrA..3..7xpF=
0x0030: 0d5f 878d 6898 9a77 04af 4708 7a3b b9b3 ...h.....G..j..
0x0040: ce29 d47f fecb bd72 e3b8 2eb4 b9a9 4033 ..).....f.....@3
0x0050: 94ad c019 04aa b3c5 7cd7 25ab 354b 10cd .....|.X.5K..
0x0060: 54c5 b544 0ca3 cf80 T..D....
14:51:40.271942 IP (tos 0x0, ttl 64, id 16252, offset 0, flags [DF], proto ESP (50), length 104)
10.10.10.1 > 10.0.0.1: ESP(spi=0xb16474e,seq=0x3), length 84
0x0000: 4500 0068 37fc 4000 3e32 dcde 0a0a 0a01 E..h?|0..2.....
0x0010: 0a00 0001 bf16 474e 0000 0003 daba 0092 .....CA.....
0x0020: 4aef 9f93 05f0 2363 41fb 0491 8386 2315 J.....#CA.....#
0x0030: 0268 f1e0 fecd de87 f8d6 9220 c077 6658 bh.....wFX
0x0040: d135 d0da 07e8 08af 0181 de1c 2728 a76b ..S.....0.....&k
0x0050: f829 4328 1b58 e01a dae2 bed6 6769 89d4 ..)C(X.....gl..
0x0060: b374 aa55 56d7 e917 ..t.UV...

0x0030: 8fde 9eeb 5a90 76dc fd60 19ad 1171 eb98 .m..Z.V.....q..
0x0040: f3be e339 893e 6273 4228 5539 5f44 62fc ...9..>ba8(U9_Db.
0x0050: c355 e015 65d1 6a4d 47ab 518c ab1c d872 ..u..e.jmG.Q....r
0x0060: f0c2 16bf e500 efc4 .....
14:51:40.271766 IP (tos 0x0, ttl 64, id 40890, offset 0, flags [DF], proto ESP (50), length 104)
10.0.0.1 > 10.10.10.1: ESP(spi=0xb16474e,seq=0x3), length 84
0x0000: 4500 0068 9fba 4000 3e32 7e9e 0a00 0001 E..h..0..>2.....
0x0010: 0a0a 0a01 a6f7 a457 0000 0004 03a6 7412 .....W.....t.
0x0020: 9ed7 8925 7226 80a0 33e0 b33f 7070 4030 ...NrA..3..7xpF=
0x0030: 0d5f 878d 6898 9a77 04af 4708 7a3b b9b3 ...h.....G..j..
0x0040: ce29 d47f fecb bd72 e3b8 2eb4 b9a9 4033 ..).....f.....@3
0x0050: 94ad c019 04aa b3c5 7cd7 25ab 354b 10cd .....|.X.5K..
0x0060: 54c5 b544 0ca3 cf80 T..D....
14:51:40.271992 IP (tos 0x0, ttl 62, id 16252, offset 0, flags [DF], proto ESP (50), length 104)
10.10.10.1 > 10.0.0.1: ESP(spi=0xb16474e,seq=0x3), length 84
0x0000: 4500 0068 37fc 4000 3e32 dcde 0a0a 0a01 E..h?|0..2.....
0x0010: 0a00 0001 bf16 474e 0000 0003 daba 0092 .....CA.....
0x0020: 4aef 9f93 05f0 2363 41fb 0491 8386 2315 J.....#CA.....#
0x0030: 0268 f1e0 fecd de87 f8d6 9220 c077 6658 bh.....wFX
0x0040: d135 d0da 07e8 08af 0181 de1c 2728 a76b ..S.....0.....&k
0x0050: f829 4328 1b58 e01a dae2 bed6 6769 89d4 ..)C(X.....gl..
0x0060: b374 aa55 56d7 e917 ..t.UV...

```

## 1.3 Comparaison

concernant la vitesse, après avoir utilisé iperf pour mesurer la différence de vitesse entre le mode transport et le mode tunnel nous remarquons que leurs vitesses sont si proches donc nous pouvons constater que le mode choisi n'affecte pas trop la vitesse de transfert.

```
vlad@vlad:~$ sudo ip netns exec h2 iperf -c 10.10.10.1
-----
Client connecting to 10.10.10.1, TCP port 5001
TCP window size: 425 KByte (default)
-----
[ 1] local 10.0.0.1 port 33998 connected with 10.10.10.1 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 1] 0.0000-10.0117 sec  331 MBytes  277 Mbits/sec
vlad@vlad:~$
```

**Tunnel**

```
vlad@vlad:~$ sudo ip netns exec h2 iperf -c 10.10.10.1
-----
Client connecting to 10.10.10.1, TCP port 5001
TCP window size: 85.0 KByte (default)
-----
[ 1] local 10.0.0.1 port 40718 connected with 10.10.10.1 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 1] 0.0000-10.0044 sec  365 MBytes  306 Mbits/sec
vlad@vlad:~$
```

**Transport**

Pour comparer les tailles des paquets des deux modes, j'ai sniffé le trafic des deux modes en utilisant tcpdump et à partir des paquets capturés, nous pouvons remarquer que le paquet en mode tunnel fait 120 octets, soit 1,15 fois plus grand que le paquet en mode transport (104 octets). cela est dû à l'encapsulation du paquet IP d'origine avec un nouvel en-tête IP en mode tunnel.

```

0x0030: 8f6d 9eeb 5a90 76dc fd60 19ad 1171 eb98  .m..Z.v..`...q..
0x0040: f5be e539 893e 6273 4228 5539 5f44 62fc  ...9.>bsB(U9_db.
0x0050: c355 e615 65d1 6a4d 47ab 518c ab1c d872  .U..e.jMG.Q....r
0x0060: f0c2 16bf e560 efcd  ....
14:51:40.271817 IP (tos 0x0, ttl 62, id 40890, offset 0, flags [DF], proto ESP (50), length 104)
10.0.0.1 > 10.10.10.1: ESP(spi=0xa6f7a457,seq=0x4), length 84
0x0000: 4500 0068 9fba 4000 3e32 7e9e 0a00 0001  E..h..@.>2~....
0x0010: 0a0a 0a01 a6f7 a457 0000 0004 03a6 7412  ....W.....t.
0x0020: 9ed7 0925 7226 80a0 33e0 b33f 7870 463d  ...%r&...3...?xpF=
0x0030: 0d5f 878d 6898 9af7 04af 47b8 7e3b b9b3  ._.h.....G.~;..
0x0040: 6268 f1e0 fecb bd72 e3b8 2eb4 b9a9 4033  .).....r.....@3
0x0050: 94ad c019 04aa b3c5 7cd7 25ab 354b 10cd  ....|.%5K..
0x0060: 54c5 b544 0ca3 cf80  T..D....
14:51:40.271942 IP (tos 0x0, ttl 64, id 16252, offset 0, flags [DF], proto ESP (50), length 104)
10.10.10.1 > 10.0.0.1: ESP(spi=0xbf16474e,seq=0x3), length 84
0x0000: 4500 0068 3f7c 4000 4032 dcdc 0a0a 0a01  E..h?|@.2.....
0x0010: 0a00 0001 bf16 474e 0000 0003 daba 0092  ....GN.....
0x0020: 4aef 9f93 05f0 2363 41fb 0491 8386 2315  J.....#CA.....#.
0x0030: 6268 f1e0 fcec de87 f8d6 9220 c677 6658  bh.....wFX
0x0040: d135 ddda 07e8 084f 0181 de1c 2726 a76b  .5.....0....'&.k
0x0050: f829 4328 1b58 e61a dae2 bed6 6769 89d4  .)C(X.....gi..
0x0060: b374 aa55 56d7 e917  .t.UV...

```

TRANSPORT

```

vlad@vlad:~$ sudo ip netns exec r1 tcpdump -lnvvx "ip"
tcpdump: listening on r1-eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
14:18:57.850278 IP (tos 0x0, ttl 62, id 0, offset 0, flags [DF], proto ESP (50), length 120)
10.0.0.1 > 10.10.10.1: ESP(spi=0x2ff84fae,seq=0x9), length 100
0x0000: 4500 0078 0000 4000 3e32 1e49 0a00 0001  E..x..@.>2.I....
0x0010: 0a0a 0a01 2ff8 4fae 0000 0009 6d48 25b1  ..../.O.....mH%.
0x0020: c6cf 3763 a27a b614 814d 84fd 9584 e76f  ..7c.z...M.....o
0x0030: 97c5 1f24 01d6 1049 30d5 d7b2 fcc7 17af  ...$.I0.....
0x0040: 89a0 8c80 7711 0fc7 2cc9 16ec c4a8 01ef  ....W.....
0x0050: 15b3 866f 01eb 3c44 6e52 2c93 e695 b4dd  ...o...<DnR,....
0x0060: 1fd0 e15b 439c 2b92 7c87 cc75 4913 af5a  ...[C.+|...uI..Z
0x0070: 6663 5f1b 7296 ff8e  fc_r...
14:18:57.850423 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ESP (50), length 120)
10.10.10.1 > 10.0.0.1: ESP(spi=0xe90fa004,seq=0x7), length 100
0x0000: 4500 0078 0000 4000 4032 1c49 0a0a 0a01  E..x..@.2.I....
0x0010: 0a00 0001 e90f a004 0000 0007 4d25 d5cf  ....M%..
0x0020: d07f 3025 59c2 5a89 dfb1 3fac f537 1a08  ..0%Y.Z....?..7..
0x0030: c502 67aa 0ac6 e660 bb4f ed3c 0e41 7eab  ..g....`.0.<.A~.
0x0040: 6b46 ec09 accf cdca cf56 8c29 c4da 0316  kF.....V.)....
0x0050: d1c5 c133 c158 16ee aa88 d1ce d50d aaa1  ...3.X.....
0x0060: 35e2 bc89 1e72 b150 320f c489 c9b6 5197  5....r.P2....Q.
0x0070: 28e9 4f64 824b 6e9f  (.Od.Kn.

```

Tunnel

## Chapitre 2

# Tunnel avec IKE

### 2.1 Configuration

Dans la partie configuration j'ai créé pour chaque namespace les paramètres utilisés dans le tunnel (clés, certificats) en appliquant les commandes présentées dans la donnée et j'ai créé le fichier de configuration swanctl.conf pour chaque namespace

```
1 #swanctl.conf h2
2     connections {
3         host-host {
4             remote_addrs = 10.10.10.1
5             local {
6                 auth=pubkey
7                 certs = h2.pem
8             }
9             remote {
10                 auth = pubkey
11                 id = "CN=10.10.10.1"
12             }
13             children {
14                 net-net {
15                     start_action = trap
16                 }
17             }
18         }
19 }
```

```
1 #swanctl.conf r2
2     connections {
3     net-net {
4         remote_addrs = 172.16.1.1
5         local {
6             auth = pubkey
7             certs = r2.pem
8         }
9         remote {
10             auth = pubkey
11             id = "CN=172.16.1.1"
12         }
13     }
```

```

13 children {
14     net-net {
15         local_ts = 10.0.0.0/24
16         remote_ts = 10.10.10.0/24
17         start_action = trap
18     }
19 }
20 }
21 }

```

## 2.2 Établissement du tunnel entre h1 et h2

Après avoir utilisé les commandes suivantes dans les namespaces h1 et h2 j'ai obtenu les résultats ci-dessous.

```

1 $ sudo ip netns exec h1 bash
2 $ sudo killall charon
3 $ sudo killall starter
4 $ sudo mkdir /tmp/h1
5 $ sudo mount --bin /tmp/h1 /run
6 $ sudo ipsec start
7 $ swanctl --load-creds
8 $ swanctl --load-conns

```

The image shows two terminal windows side-by-side, both running on a system named 'joe-VirtualBox'.

The left window is titled 'root@joe-VirtualBox: /etc/netns/h1/swanctl 68x32'. It shows the execution of the following commands:
 

```

root@joe-VirtualBox:/etc/netns/h1/swanctl# sudo swanctl --load-creds -f swanctl.conf
loaded certificate from './x509/h1.pem'
loaded certificate from './x509ca/ca-cert.pem'
opening directory './x509ocsp' failed: No such file or directory
opening directory './x509aa' failed: No such file or directory
opening directory './x509ac' failed: No such file or directory
opening directory './x509crl' failed: No such file or directory
opening directory './pubkey' failed: No such file or directory
loaded RSA key from './private/h1Key.pem'
opening directory './rsa' failed: No such file or directory
opening directory './ecdsa' failed: No such file or directory
opening directory './bliss' failed: No such file or directory
opening directory './pkcs8' failed: No such file or directory
opening directory './pkcs12' failed: No such file or directory
root@joe-VirtualBox:/etc/netns/h1/swanctl# sudo swanctl --load-conns -f swanctl.conf
loaded connection 'host-host'
successfully loaded 1 connections, 0 unloaded
root@joe-VirtualBox:/etc/netns/h1/swanctl#
    
```

The right window is titled 'root@joe-VirtualBox: /etc/netns/h2/swanctl 72x32'. It shows the execution of the following commands:
 

```

root@joe-VirtualBox:/etc/netns/h2/swanctl# sudo swanctl --load-creds -f swanctl.conf
loaded certificate from './x509/h2.pem'
loaded certificate from './x509ca/ca-cert.pem'
opening directory './x509ocsp' failed: No such file or directory
opening directory './x509aa' failed: No such file or directory
opening directory './x509ac' failed: No such file or directory
opening directory './x509crl' failed: No such file or directory
opening directory './pubkey' failed: No such file or directory
loaded RSA key from './private/h2Key.pem'
opening directory './rsa' failed: No such file or directory
opening directory './ecdsa' failed: No such file or directory
opening directory './bliss' failed: No such file or directory
opening directory './pkcs8' failed: No such file or directory
opening directory './pkcs12' failed: No such file or directory
root@joe-VirtualBox:/etc/netns/h2/swanctl# sudo swanctl --load-conns -f swanctl.conf
loaded connection 'host-host'
successfully loaded 1 connections, 0 unloaded
root@joe-VirtualBox:/etc/netns/h2/swanctl#
    
```

At the bottom right of the terminal windows, there is a watermark that says 'Activate Windows Go to Settings to activate Windows.'



Ensuite, j'ai envoyé un ping de h1 à h2, nous pouvons remarquer que l'icmp-seq a commencé par 2, ce qui indique que le premier paquet icmp a été perdu car le tunnel n'était pas encore établi. Dans l'autre partie de la capture, la commande swanctl -log m'a montré le succès de la négociation entre h1 et h2

The image shows two terminal windows from a VirtualBox environment. The left window, titled 'root@joe-VirtualBox: /etc/netns/h1/swanctl 66x32', shows the output of a 'ping 10.0.0.1' command. It displays five successful ping requests with varying sequence numbers (2 to 6) and times, indicating that the first packet was lost. The right window, titled 'root@joe-VirtualBox: /etc/netns/h2/swanctl 74x32', shows the output of 'swanctl -log'. It details the IKE negotiation process between the two hosts, including certificate selection, authentication, and the establishment of the 'host-host' IKE SA and 'CHILD\_SA net-net(2)'.

```

root@joe-VirtualBox: /etc/netns/h1/swanctl 66x32
root@joe-VirtualBox: /etc/netns/h1/swanctl# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data:
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.202 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.206 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.315 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=64 time=0.241 ms
^C
--- 10.0.0.1 ping statistics ---
5 packets transmitted, 4 received, 20% packet loss, time 4080ms
rtt min/avg/max/mdev = 0.202/0.241/0.315/0.045 ms
root@joe-VirtualBox: /etc/netns/h1/swanctl#

root@joe-VirtualBox: /etc/netns/h2/swanctl 74x32
05[IKE] received end entity cert "CN=10.10.10.1"
05[CFG] looking for peer configs matching 10.0.0.1[CN=10.0.0.1]...10.10.10.1[CN=10.10.10.1]
05[CFG] selected peer config 'host-host'
05[CFG] using certificate "CN=10.10.10.1"
05[CFG] using trusted ca certificate "CN=VPN root CA"
05[CFG] reached self-signed root ca with a path length of 0
05[CFG] checking certificate status of "CN=10.10.10.1"
05[CFG] certificate status is not available
05[IKE] authentication of 'CN=10.10.10.1' with RSA_EMSA_PKCS1_SHA2_384 successful
05[IKE] peer supports MOBIKE
05[IKE] authentication of 'CN=10.0.0.1' (myself) with RSA_EMSA_PKCS1_SHA2_384 successful
05[IKE] IKE_SA host-host[1] established between 10.0.0.1[CN=10.0.0.1]...10.10.10.1[CN=10.10.10.1]
05[IKE] scheduling rekeying in 12987s
05[IKE] maximum IKE_SA lifetime 14427s
05[IKE] sending end entity cert "CN=10.0.0.1"
05[CFG] selected proposal: ESP:AES_GCM_16_128
05[IKE] CHILD_SA net-net(2) established with SPIs cd6fc87c_i cf3177e2_o and TS 10.0.0.1/32 == 10.10.10.1/32
05[ENC] generating IKE_AUTH response 1 [ IDr CERT AUTH SA Tst Isr N(MOBIKE_SUP) N(NO_ADD_ADDR) ]
05[ENC] splitting IKE message (2000 bytes) into 2 fragments
05[ENC] generating IKE_AUTH response 1 [ EF(1/2) ]
05[ENC] generating IKE_AUTH response 1 [ EF(2/2) ]
05[NET] sending packet: from 10.0.0.1[4500] to 10.10.10.1[4500] (1236 bytes)
05[NET] sending packet: from 10.0.0.1[4500] to 10.10.10.1[4500] (836 bytes)

```

## 2.3 Établissement du tunnel entre r1 et r2

Pour r1 et r2, j'ai exécuté les mêmes commandes précédentes.

The image shows two terminal windows. The left window, titled 'root@joe-VirtualBox: /etc/netns/r2/swanctl 71x32', shows the output of 'sudo swanctl --load-creds -f swanctl.conf' for the r2 namespace. It lists various certificates and keys that were loaded successfully, while some directories (like /x509ocsp, /x509aa, /x509ac, /x509crl, /pubkey) failed to be loaded because they do not exist. The right window, titled 'root@joe-VirtualBox: /etc/netns/r1/swanctl 69x32', shows the same command being executed for the r1 namespace, with similar results: certificates and keys are loaded, but some directories are missing.

```

root@joe-VirtualBox: /etc/netns/r2/swanctl 71x32
root@joe-VirtualBox: /etc/netns/r2/swanctl# sudo swanctl --load-creds -f swanctl.conf
loaded certificate from './x509/r2.pem'
loaded certificate from './x509ca/ca-cert.pem'
opening directory './x509ocsp' failed: No such file or directory
opening directory './x509aa' failed: No such file or directory
opening directory './x509ac' failed: No such file or directory
opening directory './x509crl' failed: No such file or directory
opening directory './pubkey' failed: No such file or directory
loaded RSA key from './private/r2Key.pem'
opening directory './rsa' failed: No such file or directory
opening directory './ecdsa' failed: No such file or directory
opening directory './bliss' failed: No such file or directory
opening directory './pkcs8' failed: No such file or directory
opening directory './pkcs12' failed: No such file or directory
root@joe-VirtualBox: /etc/netns/r2/swanctl# sudo swanctl --load-conns -f swanctl.conf
loaded connection 'net-net'
successfully loaded 1 connections, 0 unloaded
root@joe-VirtualBox: /etc/netns/r2/swanctl#

root@joe-VirtualBox: /etc/netns/r1/swanctl 69x32
root@joe-VirtualBox: /etc/netns/r1/swanctl# sudo swanctl --load-creds -f swanctl.conf
loaded certificate from './x509/r1.pem'
loaded certificate from './x509ca/ca-cert.pem'
opening directory './x509ocsp' failed: No such file or directory
opening directory './x509aa' failed: No such file or directory
opening directory './x509ac' failed: No such file or directory
opening directory './x509crl' failed: No such file or directory
opening directory './pubkey' failed: No such file or directory
loaded RSA key from './private/r1Key.pem'
opening directory './rsa' failed: No such file or directory
opening directory './ecdsa' failed: No such file or directory
opening directory './bliss' failed: No such file or directory
opening directory './pkcs8' failed: No such file or directory
opening directory './pkcs12' failed: No such file or directory
root@joe-VirtualBox: /etc/netns/r1/swanctl# sudo swanctl --load-conns -f swanctl.conf
loaded connection 'net-net'
successfully loaded 1 connections, 0 unloaded
root@joe-VirtualBox: /etc/netns/r1/swanctl#

```

Ensuite, j'ai pingé de h1 à h2 et nous pouvons remarquer sur la capture ci-dessous que le ping arrive à destination après avoir perdu le premier paquet icmp, ce qui indique que la négociation a réussi et que les tunnels ont été combinés.

The image shows two terminal windows from a virtual machine named 'joe-VirtualBox'.

The left window shows the output of a ping command from 'h1' to '10.0.0.1'. The output indicates a 10% packet loss (9 out of 10 packets received) and a round-trip time of approximately 0.2 ms. The command used was 'ping 10.0.0.1'.

The right window shows the output of the 'swanctl' daemon logs. It details the IKE negotiation process, including the establishment of the IKE SA, the selection of the peer configuration 'net-net', and the successful authentication of the peer using RSA and SHA256. The logs also show the establishment of the CHILD\_SA and the generation of the IKE\_AUTH response.

## 2.4 Résultats

### 2.4.1 h1 et h2

Avant la construction du tunnel entre r1 et r2 j'ai testé le tunnel entre h1 et h2 en établissant une connexion TCP entre les deux en utilisant socat. Ensuite, j'ai envoyé un message de h1 à h2 qui va être sniffé à l'aide de Tcpdump. Nous pouvons remarquer avec la capture ci-dessous que le payload est chiffré et que le protocole utilisé est ESP.

The image shows three terminal windows from a virtual machine named 'joe-VirtualBox'.

The top-left window shows the output of the 'socat' command, which is used to establish a TCP connection between 'h1' and 'h2' on port 6789.

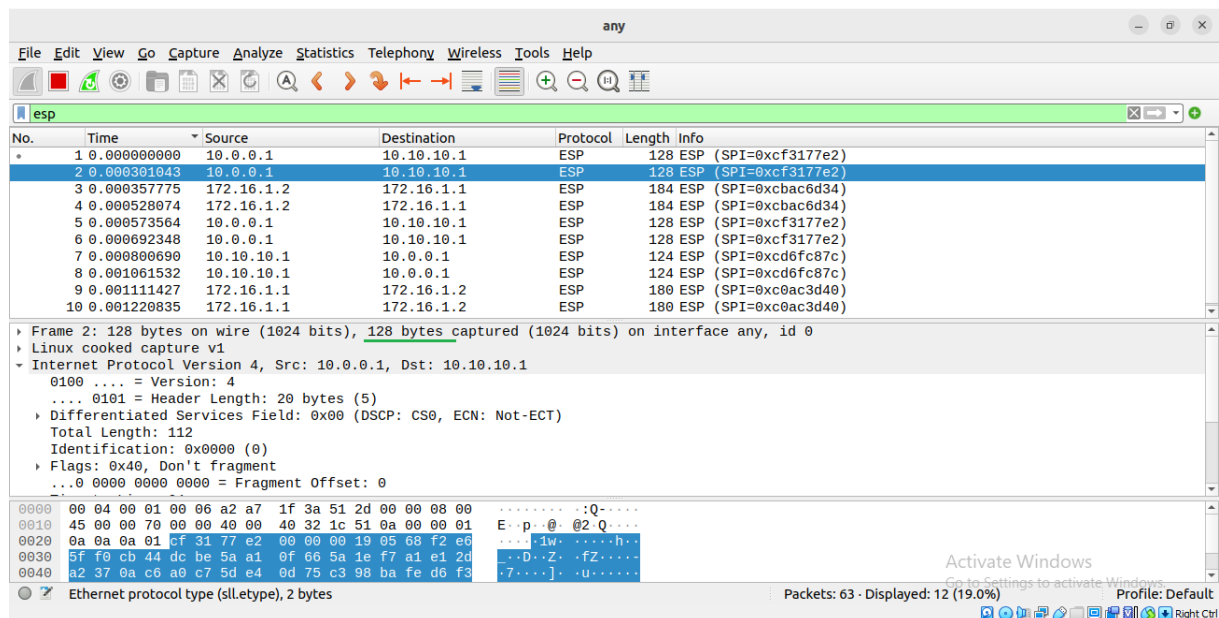
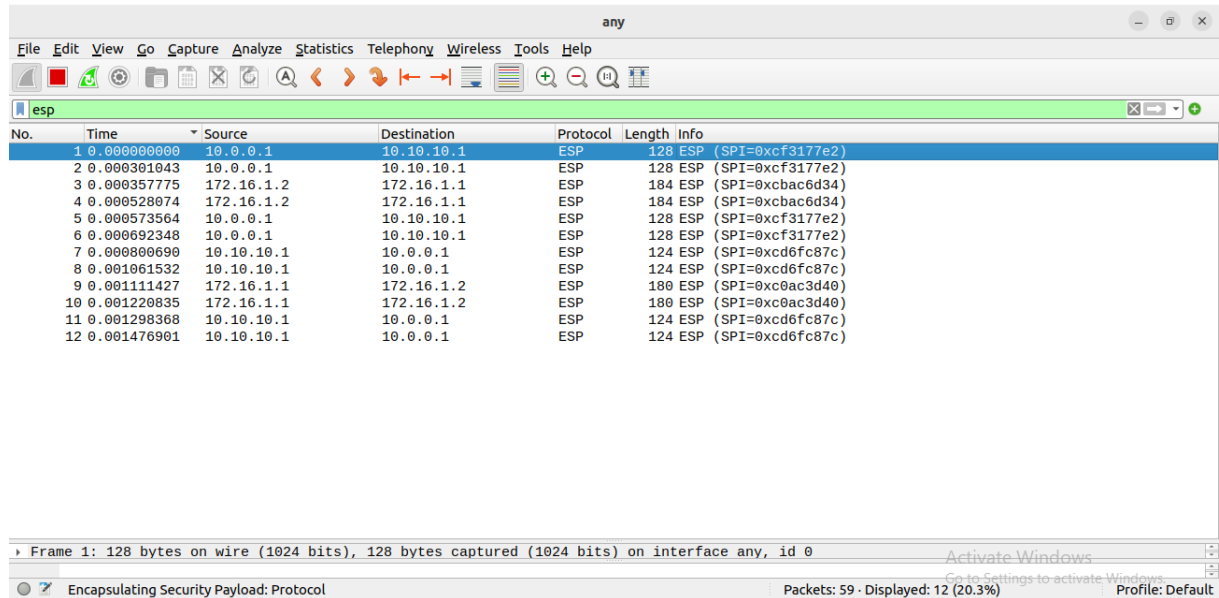
The top-right window shows the output of the 'socat' command, which is used to listen for incoming connections on port 6789.

The bottom window shows the output of the 'tcpdump' command, which is used to capture network traffic. The output shows a packet from 'h1' to 'h2' with a payload of type ESP (Encapsulating Security Protocol), indicating that the data is encrypted.

## 2.4.2 r1 et r2

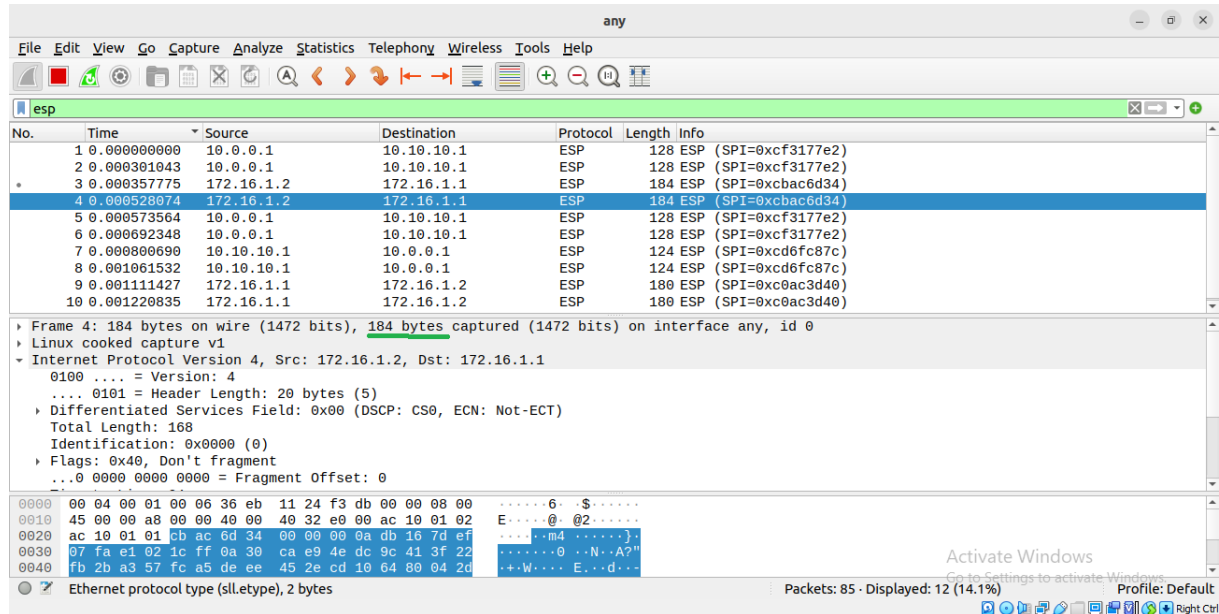
Maintenant, après l'établissement des tunnels entre r1 et r2, j'ai utilisé Wireshark pour capturer le trafic entre h1 et h2.

Dans les deux captures suivantes, nous pouvons voir qu'entre h1 et r1, le payload du paquet est chiffré avec le protocole ESP sans aucun changement dans l'en-tête IP.



Dans la capture ci-dessous nous pouvons remarquer que le paquet ip d'origine a été encapsulé avec un nouvel en-tête ip ayant l'ip source 172.16.1.2 (adresse r2) et l'ip destination 172.168.1.1 (adresse r1) et chiffré avec le protocole ESP.

De plus, la longueur du paquet est devenue plus grande après l'encapsulation de 184 octets, soit 1,43 fois plus grande que le paquet avant l'encapsulation (128 octets).



## 2.5 RoadWarrior

### 2.5.1 configuration

J'ai créé les mêmes paramètres que les configurations précédentes pour h1 et r2 (clés et certificats). J'ai uniquement modifié le champ CN du certificat des namespaces en mettant le mail à la place de l'adresse ip.

La création des certificats est devenue la suivante :

```
1 #namespace r2 certificate
2 pki --pub --in pki/private/r2Key.pem --type rsa | pki --issue --cacert pki/
  cacerts/ca-cert.pem --cakey pki/private/ca-key.pem --lifetime 1825 --dn "CN=
  r2@assr.com" --outform pem > pki/certs/r2.pem

1 #namespace h1 certificate
2 pki --pub --in pki/private/h1Key.pem --type rsa | pki --issue --cacert pki/
  cacerts/ca-cert.pem --cakey pki/private/ca-key.pem --lifetime 1825 --dn "CN=
  h1@assr.com" --outform pem > pki/certs/h1.pem
```

Concernant les fichiers de swanctl.conf le contenu de chaque namespace est :

```
1 #r2 swanctl.conf
2 connections {
3     rw {
4         local {
5             auth=pubkey
6             certs=r2.pem
7         }
8         remote {
9             auth=pubkey
10        }
11        children {
12            rw {
13                local_ts=10.0.0.0/24
14            }
15        }
16    }
17 }
```

```
1 #h1 swanctl.conf
2 connections {
3     home {
4         remote_addrs = 172.16.1.2
5         local {
6             auth=pubkey
7             certs=h1.pem
8         }
9         remote {
10            auth=pubkey
11            id="CN=r2@assr.com"
12        }
13        children {
14            home {
15                remote_ts=10.0.0.0/24
16                start_action=start
17            }
18        }
19    }
20 }
```

Ensuite, j'ai chargé le certificat et la connexion pour chaque namespace :

```

root@joe-VirtualBox: /etc/netns/r2/swanctl 67x31
root@joe-VirtualBox:/etc/netns/r2/swanctl# sudo swanctl --load-creds -f swanctl.conf
loaded certificate from './x509/r2.pem'
loaded certificate from './x509ca/ca-cert.pem'
opening directory './x509ocsp' failed: No such file or directory
opening directory './x509aa' failed: No such file or directory
opening directory './x509ac' failed: No such file or directory
opening directory './x509crl' failed: No such file or directory
opening directory './pubkey' failed: No such file or directory
loaded RSA key from './private/r2Key.pem'
opening directory './rsa' failed: No such file or directory
opening directory './ecdsa' failed: No such file or directory
opening directory './bliss' failed: No such file or directory
opening directory './pkcs8' failed: No such file or directory
opening directory './pkcs12' failed: No such file or directory
root@joe-VirtualBox:/etc/netns/r2/swanctl# sudo swanctl --load-conn s -f swanctl.conf
loaded connection 'rw'
successfully loaded 1 connections, 0 unloaded
root@joe-VirtualBox:/etc/netns/r2/swanctl#

root@joe-VirtualBox: /etc/netns/h1/swanctl 73x32
root@joe-VirtualBox:/etc/netns/h1/swanctl# sudo swanctl --load-creds -f swanctl.conf
loaded certificate from './x509/h1.pem'
loaded certificate from './x509ca/ca-cert.pem'
opening directory './x509ocsp' failed: No such file or directory
opening directory './x509aa' failed: No such file or directory
opening directory './x509ac' failed: No such file or directory
opening directory './x509crl' failed: No such file or directory
opening directory './pubkey' failed: No such file or directory
loaded RSA key from './private/h1Key.pem'
opening directory './rsa' failed: No such file or directory
opening directory './ecdsa' failed: No such file or directory
opening directory './bliss' failed: No such file or directory
opening directory './pkcs8' failed: No such file or directory
opening directory './pkcs12' failed: No such file or directory
root@joe-VirtualBox:/etc/netns/h1/swanctl# sudo swanctl --load-conn s -f swanctl.conf
loaded connection 'home'
successfully loaded 1 connections, 0 unloaded
root@joe-VirtualBox:/etc/netns/h1/swanctl#

```

## 2.5.2 Résultat

La capture suivante nous montre des paquets ISAKMP capturés avec Wireshark, ce qui indique que les namespaces ont négocié avec succès les uns avec les autres sur l'établissement du tunnel.

No.	Time	Source	Destination	Protocol	Length	Info
30	54.616195778	10.10.10.1	172.16.1.2	ISAKMP	948	IKE_SA_INIT MID=00 Initiator Request
31	54.616264064	10.10.10.1	172.16.1.2	ISAKMP	948	IKE_SA_INIT MID=00 Initiator Request
36	54.616425623	10.10.10.1	172.16.1.2	ISAKMP	948	IKE_SA_INIT MID=00 Initiator Request
37	54.616484248	10.10.10.1	172.16.1.2	ISAKMP	948	IKE_SA_INIT MID=00 Initiator Request
38	54.617287730	172.16.1.2	10.10.10.1	ISAKMP	80	IKE_SA_INIT MID=00 Responder Response
39	54.617390409	172.16.1.2	10.10.10.1	ISAKMP	80	IKE_SA_INIT MID=00 Responder Response
40	54.617400464	172.16.1.2	10.10.10.1	ISAKMP	80	IKE_SA_INIT MID=00 Responder Response
41	54.617510539	172.16.1.2	10.10.10.1	ISAKMP	80	IKE_SA_INIT MID=00 Responder Response

Frame 30: 948 bytes on wire (7584 bits), 948 bytes captured (7584 bits) on interface any, id 0

```

0000  00 03 00 01 00 06 e2 da 26 ae 5d 8a 00 00 08 00  .....&.]....
0010  45 00 03 a4 63 1e 40 00 40 11 13 0e 0a 0a 0a 01  E...C-@. @.
0020  ac 10 01 02 01 f4 01 f4 03 90 c4 be 03 dd b5 b5  ..).....! ".
0030  8b b5 29 83 00 00 00 00 00 00 00 00 21 20 22 08  ..).....! ".
0040  00 00 00 00 00 00 00 00 22 00 02 c8 02 00 01 44  .....D
0050  01 01 00 23 03 00 00 0c 01 00 00 0c 80 0e 00 80  ...#.....
0060  03 00 00 0c 01 00 00 0c 80 0e 00 c0 03 00 00 0c  .....
0070  01 00 00 0c 80 0e 01 00 03 00 00 0c 01 00 00 0d  .....
0080  80 0e 00 80 03 00 00 0c 01 00 00 0d 80 0e 00 c0  .....
0090  03 00 00 0c 01 00 00 0d 80 0e 01 00 03 00 00 0c  .....
00a0  01 00 00 17 80 0e 00 80 03 00 00 0c 01 00 00 17  ....

```

Et dans la capture ci-dessous, j'ai testé le tunnel entre les namespaces h1 et h2 avec socat en utilisant tcpdump dans r1 pour sniffer le trafic. Nous voyons que le trafic a été chiffré avec ESP ce qui indique que le tunnel est bien établi.

The screenshot shows a network capture in a terminal window. The left pane displays the output of `tcpdump` on interface `r1-eth0`. A packet from `10.10.10.1` to `172.16.1.2` is highlighted with a green box. The packet details show it is an ESP (Encapsulating Security Protocol) packet with a length of 92 bytes. A green arrow points to the packet data, which is labeled "Encrypted DATA". The right pane shows the output of `cat` on interface `h1-swanc1`, displaying the received data. The terminal window title is `root@joe-VirtualBox: /etc/netns/r1 195x32`.

```

root@joe-VirtualBox: /etc/netns/r1 195x32
root@joe-VirtualBox: /etc/netns/r1# sudo tcpdump -lnvX 'ip'
tcpdump: listening on r1-eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
20:30:08.789632 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ESP (50), length 112)
10.10.10.1 > 172.16.1.2: ESP(spi=0xce7354a4,seq=0x3), length 92
0x0000: 4500 0070 0000 4000 4032 793f 0a0a 0a01 E..p..@.zy?....
0x0010: ac10 0102 ce73 54a4 0000 0003 8c1f 7ec0 .....ST.....
0x0020: 395d 49b6 6a7a 24c7 1398 3e73 afa3 3a82 9]I.jz$...>s...
0x0030: f583 9051 d7f7 818b dedc 8371 0130 9973 ...Q.....q.0.s
0x0040: 0d61 eed7 21a2 6493 e8b1 4006 c8d2 d4e9 .a..l.d...@....
0x0050: 1ae2 03e5 ec26 e7a1 3b2b 6a16 c011 fc38 .....&...;+j...8
0x0060: e5cd 476d ff4d 26a1 e146 528d 9b58 aa1c ..Gm.M&..FR..X..
20:30:08.790336 IP (tos 0x0, ttl 63, id 0, offset 0, flags [DF], proto ESP (50), length 108)
172.16.1.2 > 10.10.10.1: ESP(spi=0xcac79094,seq=0x2), length 88
0x0000: 4500 006c 0000 4000 3f32 7a43 ac10 0102 E..l..@.72zC....
0x0010: 0a0a 0a01 cac7 9094 0000 0002 9e80 03d3 .....
0x0020: 054e cb23 4cf4 6ebe e279 4dc7 6b07 e19f ..N.#L.n..yM.k...
0x0030: 940e 52c0 71c4 2389 6913 2cad 0964 45bb ..R.q.#.l,..dE..
0x0040: 8f6c b13f f391 9040 4b08 e13c d562 ddc9 .l.?...@K...<.b..
0x0050: 63c3 cb31 69de 96ab 053e 96dc a974 e6b1 c..il.....>...t..
0x0060: 0eb4 3500 8287 958a 29c6 4244 ..5.....).BD

```