Anno Accademico 2021/2022

# Laboratorio di Internet


Politecnico di Torino

# Report I

*Connectivity Check - Advanced Ping*

Gruppo 14

Andreas Brummer (s270332)

Alessandro Ciullo (s269589)

Andrea Scamporrino (s270971)

# 1 Configurazione di rete

Nell'effettuare i test é stata usata la seguente configurazione di rete:

| | |
|---|---|
| Indirizzo di rete | 172.16.14.0 |
| Indirizzo di broadcast | 172.16.14.63 |
| NetMask | 255.255.255.192 |

| Nome host | Indirizzo IP |
|---|---|
| H1 | 172.16.14.1/26 |
| H2 | 172.16.14.2/26 |
| H3 | 172.16.14.3/26 |

Tabella 1.1: Configurazione di rete

# 2 Ping agli indirizzi di broadcast e di rete

In questa sezione osserviamo e analizziamo il comportamento di una rete locale e degli host che la compongono quando i pacchetti, destinati all'indirizzo di broadcast o di rete, vengono trasmessi sul canale fisico.

## 2.1 Ping all'indirizzo broadcast

**-Cosa succede quando un host pinga l'indirizzo di broadcast?**

In questo caso abbiamo lanciato il comando "ping 172.16.14.63 -b -c 6" con l'host H1. Dall'immagine

```
laboratorio@laboratorio:~$ ping 172.16.14.63 -b -c 6
WARNING: pinging broadcast address
PING 172.16.14.63 (172.16.14.63) 56(84) bytes of data.
64 bytes from 172.16.14.1: icmp_seq=1 ttl=64 time=0.037 ms
64 bytes from 172.16.14.2: icmp_seq=1 ttl=64 time=1.33 ms (DUP!)
64 bytes from 172.16.14.3: icmp_seq=1 ttl=64 time=1.65 ms (DUP!)
64 bytes from 172.16.14.1: icmp_seq=2 ttl=64 time=0.045 ms
64 bytes from 172.16.14.2: icmp_seq=2 ttl=64 time=0.702 ms (DUP!)
64 bytes from 172.16.14.3: icmp_seq=2 ttl=64 time=0.964 ms (DUP!)
64 bytes from 172.16.14.1: icmp_seq=3 ttl=64 time=0.044 ms
64 bytes from 172.16.14.2: icmp_seq=3 ttl=64 time=0.723 ms (DUP!)
64 bytes from 172.16.14.3: icmp_seq=3 ttl=64 time=1.65 ms (DUP!)
64 bytes from 172.16.14.1: icmp_seq=4 ttl=64 time=0.043 ms
64 bytes from 172.16.14.2: icmp_seq=4 ttl=64 time=0.740 ms (DUP!)
64 bytes from 172.16.14.3: icmp_seq=4 ttl=64 time=1.66 ms (DUP!)
64 bytes from 172.16.14.1: icmp_seq=5 ttl=64 time=0.043 ms
64 bytes from 172.16.14.2: icmp_seq=5 ttl=64 time=0.709 ms (DUP!)
64 bytes from 172.16.14.3: icmp_seq=5 ttl=64 time=1.60 ms (DUP!)
64 bytes from 172.16.14.1: icmp_seq=6 ttl=64 time=0.044 ms

--- 172.16.14.63 ping statistics ---
6 packets transmitted, 6 received, +10 duplicates, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 0.037/0.748/1.659/0.639 ms
```

Figura 2.1: Ping all'indirizzo broadcast

2.1 possiamo notare che vengono ricevute tre risposte di cui una proveniente dall'interfaccia di loopback e due dagli altri host della rete (H2 ed H3). La risposta che proviene dall'interfaccia di loopback non dovendo scendere per tutta la pila protocollare é quella che arriva più velocemente, mentre le altre due risposte sono più lente e sono segnate come duplicate dalla scritta "**DUP!**".

L'applicazione termina alla ricezione del primo pacchetto con numero di sequenza 6 (gli ultimi due duplicati sono quindi scartati).

Avvalendoci del software "**Wireshark**" possiamo notare come ad inviare le ARP request siano soltanto H2 ed H3. H1 infatti, trasmettendo in broadcast, sa a priori che l'indirizzo MAC di destinazione é **FF:FF:FF:FF:FF:FF** come mostrato in appendice nella figura 5.1.

Al termine dell'esperienza le ARP table di H2 e H3 contengono soltanto l'indirizzo di H1; l'ARP table di H1 contiene, invece, gli indirizzi di entrambi gli host H2 ed H3.

### 2.2 Ping all'indirizzo di rete

**-Cosa succede quando un host pinga l'indirizzo di rete?**

Il comportamento é analogo a quello che abbiamo visto nel punto precedente. Semanticamente infatti pingare l'indirizzo di broadcast e pingare l'intera rete hanno lo stesso significato.

Alcuni sistemi operativi, come ad esempio BSD, hanno un comportamento particolare e consentono di configurare gli host anche sull'indirizzo di rete stesso (questa possibilità crea conflitti nel momento in cui si viene ad interagire con sistemi operativi diversi).

## 3 Ping con indirizzi IP duplicati

In questa sezione studiamo il comportamento di una LAN in cui due degli host vengono configurati erroneamente con lo stesso indirizzo IP. Da qui in poi faremo riferimento a questi due host come H1 e H1', mentre al terzo host, configurato correttamente, come H2.

| | |
|---|---|
| H1 | 172.16.14.1/26 |
| H1' | 172.16.14.1/26 |
| H2 | 172.16.14.2/26 |

Tabella 3.1: Configurazione con IP duplicati

### 3.1 Ping da H2 ad H1

**-Cosa succede quando H2 pinga H1? Controllare le tabelle ARP di H1, H1' e H2.**

Quando H2 pinga H1, il protocollo ARP viene invocato per recuperare l'indirizzo MAC di H1. Tale richiesta viene inoltrata mediante un messaggio broadcast a cui rispondono entrambi gli host H1 e H1' poiché si identificano con lo stesso indirizzo IP. L'ordine in cui queste risposte vengono ricevute da H2 é perciò cruciale in quanto solo la prima viene considerata come lecita, mentre la seconda viene scartata poiché potenzialmente inaffidabile e/o dannosa. Comincia quindi lo scambio di pacchetti ICMP tra H2 e l'host che ha risposto per primo all'ARP request. I vari messaggi ARP scambiati per verifica e aggiornamenti delle ARP table vengono inviati in unicast tra i due host. L'intera situazione é mostrata in appendice in figura 5.2.

Il terzo host, rimasto escluso dalle comunicazioni, non invia più alcun pacchetto. Non partecipando a nessuno scambio di messaggi, infatti, non trasmette neanche pacchetti per l' aggiornamento e la verifica della ARP table.

Al termine del test l'ARP table di H2 contiene soltanto l'indirizzo del primo host che ha risposto alla sua ARP request; le ARP table di H1 e H1' contengono entrambe l'indirizzo di H2.

### 3.2 Ping da H1 e H1' ad H2

**-Cosa succede quando H1 e H1' pingano H2? Controllare le tabelle di ARP.**

Quando H1 e H1' pingano contemporaneamente il secondo host due ARP request vengono trasmesse in broadcast per recuperare l'indirizzo MAC di H2. Al ricevimento della prima ARP request H2 inoltra la sua ARP reply all'host sorgente della richiesta, stabilendo con esso una comunicazione. I due host continuano a

scambiarsi pacchetti ICMP fin quando non arriva ad H2 l'ARP request dell'altro host. Questa richiesta viene considerata lecita (questa falla é alla base di una forma di attacco informatico chiamato ARP Spoofing): H2 aggiorna quindi la sua ARP table e inoltra una ARP reply a questo "nuovo" host. Da questo momento in poi H2 riceve echo Request da entrambi gli host (poiché adesso sia H1 che H1' conoscono il suo MAC address) e genera una echo Reply per ciascuna di esse, ma le inoltra tutte all'host con cui più recentemente ha scambiato pacchetti ARP (come mostrato in figura 5.3); ciò causa l'invio di un messaggio fuori sequenza ad ogni aggiornamento dell'ARP table.

L'host che riceve le echo Reply scarta le risposte inattese in quanto, tendenzialmente, nessun processo corrisponde all'identificatore contenuto nell'header ICMP.

Durante il test l'ARP table di H2 muta costantemente, alternando gli indirizzi MAC di H1 e H1'. Le ARP table di H1 e H1' contengono l'indirizzo di H2.

# 4 Ping con maschere di rete errate

In questa sezione modifichiamo la maschera di rete degli host H1 ed H2 facendo in modo che:

- H1 sia in una rete più grande (126 indirizzi) che contiene anche H2
- H2 sia in una rete più piccola (62 indirizzi) che non contiene H1

Nello specifico la nuova configurazione é la seguente:

| | |
|---|---|
| H1 | 172.16.14.65/25 |
| H2 | 172.16.14.2/26 |

Tabella 4.1: Configurazione con netmask errate

## 4.1 Ping da H1 ad H2

**-Cosa succede quando H1 pinga H2? Quali pacchetti sta mandando H1? Quali pacchetti sta mandando H2? Come cambiano le tabelle ARP di H1 e H2?**

Quando H1 effettua un ping verso H2 non riceve alcuna risposta ed il terminale mostra a video il messaggio "**Destination host Unreachable**" (figura 4.1). In questo caso ARP decide di non creare il collegamento punto-punto non rispondendo alle ARP Request di un IP che si trova al di fuori dalla propria rete. Se avvenisse il contrario H2 si ritroverebbe a ricevere pacchetti IP a cui non sarebbe in grado di rispondere poiché privo di un instradamento verso H1.

ARP é quindi in grado di effettuare alcuni controlli sui pacchetti che riceve e fare verifiche sugli IP e le netmask.

Al termine del test la ARP table di H1 è incompleta e quella di H2 vuota poiché il secondo host ha scartato tutte le ARP request ricevute.

```
laboratorio@laboratorio:~$ ping 172.16.14.2
PING 172.16.14.2 (172.16.14.2) 56(84) bytes of data.
From 172.16.14.65 icmp_seq=1 Destination Host Unreachable
From 172.16.14.65 icmp_seq=2 Destination Host Unreachable
```

Figura 4.1: Messaggio ricevuto dall'host H1 nel Test 4.1

## 4.2 Ping da H2 ad H1

In questo caso invece la comunicazione si ferma a livello IP: H2 nota che l'indirizzo di destinazione é in una rete esterna verso cui non ha un instradamento e perciò non procede alla trasmissione di nessun pacchetto. Il terminale mostra quindi a video il messaggio "**Network is unreachable**" (figura 4.2).

Al termine del test le ARP table di H1 e H2 sono entrambe vuote poiché nessuna trasmissione è avvenuta.

```
laboratorio@laboratorio:~$ ping 172.16.14.65
ping: connect: Network is unreachable
```

Figura 4.2: Messaggio mostrato sul terminale H2 nel test 4.2

# 5 Netmask errate e conflitto con l'indirizzo di broadcast

In questa sezione ci occupiamo dello studio di una rete LAN quando, per via di una configurazione errata, a uno dei due host, facente parte di una data rete, viene associato l'indirizzo di broadcast di una sotto-rete alla quale appartiene il secondo host.

Abbiamo configurato la LAN nel seguente modo:

| | |
|---|---|
| H1 | 172.16.14.127/24 |
| H2 | 172.16.14.1/25 |

Tabella 5.1: Configurazione con netmask errate e conflitto con l'indirizzo di broadcast

## 5.1 Ping da H1 ad H2

**-Cosa succede quando H1 pinga H2? Quali pacchetti sta mandando H1? Quali pacchetti sta mandando H2? Come cambiano le tabelle ARP di H1 e H2?**

Quando H1 pinga H2, la prima cosa che avviene é l'inoltro di ARP request per ottenere il MAC address di H2. Le richieste una volta arrivate ad H2 vengono passate ad ARP che, poiché capace di leggere e interpretare la netmask in vigore sulla scheda di rete, si accorge di star ricevendo delle ARP request dall'indirizzo di broadcast che violano chiaramente la semantica (impossibile ricevere messaggi da "tutti") e perciò le ignora senza generare alcuna risposta. Anche questa volta, analogamente al test 4.1, il messaggio mostrato a video è "**Destination host Unreachable**".

Come conseguenza del comportamento osservato nel paragrafo precedente, al termine del test la ARP table di H1 è incompleta e quella di H2 vuota.

## 5.2 Ping da H2 ad H1

**-Cosa succede quando H2 pinga H1? Quali pacchetti sta mandando H1? Quali pacchetti sta mandando H2? Come cambiano le tabelle ARP di H1 e H2?**

Quando H2 pinga H1, ovvero l'indirizzo di broadcast dal punto di vista di H2, proprio come nell'esperienza svolta precedentemente, il primo pacchetto scambiato é una ICMP Echo Request destinato a tutti gli host sulla rete. H1 alla ricezione della richiesta trasmette delle ARP Request per ottenere il MAC address di H2. H1 non ha infatti ricevuto alcun pacchetto ARP da H2 e quindi non ne conosce l'indirizzo ethernet.

Come nell'esperienza precedente, H2 scarta tutte le ARP request per via della violazione della semantica. Si rimane perciò incastrati in questo ciclo nel quale H1 non riceve risposte alle sue ARP request e di conseguenza H2 non riceve riscontri, da parte di H1, alle sue Echo Request come mostrato in figura 5.4. Le uniche risposte mostrate sul terminale sono quelle ricevute dall'interfaccia di loopback e non sono pertanto presenti duplicati.

Anche questa volta, al termine del test, l'ARP table di H1 è incompleta in quanto il primo host non ha mai ricevuto risposte alle sue ARP request mentre l'ARP table di H2 è vuota poiché da parte del secondo host sono stati trasmessi soltanto pacchetti verso l'indirizzo di broadcast.

# Appendice

arp or icmp

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 13 | 8.413097638 | 172.16.14.1 | 172.16.14.63 | ICMP | 98 | Echo (ping) request id=0x0004, seq=1/256, ttl=64 (no respons… |
| 14 | 8.413779335 | HewlettP_15:79:be | Broadcast | ARP | 60 | Who has 172.16.14.1? Tell 172.16.14.3 |
| 15 | 8.413808501 | HewlettP_15:79:66 | HewlettP_15:79:be | ARP | 42 | 172.16.14.1 is at f4:39:09:15:79:66 |
| 16 | 8.414078796 | HewlettP_6a:29:4b | Broadcast | ARP | 60 | Who has 172.16.14.1? Tell 172.16.14.2 |
| 17 | 8.414099704 | HewlettP_15:79:66 | HewlettP_6a:29:4b | ARP | 42 | 172.16.14.1 is at f4:39:09:15:79:66 |
| 18 | 8.414372864 | 172.16.14.3 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=1/256, ttl=64 |
| 19 | 8.414965288 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=1/256, ttl=64 |
| 20 | 9.414222460 | 172.16.14.1 | 172.16.14.63 | ICMP | 98 | Echo (ping) request id=0x0004, seq=2/512, ttl=64 (no respons… |
| 21 | 9.414874326 | 172.16.14.3 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=2/512, ttl=64 |
| 22 | 9.415807446 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=2/512, ttl=64 |
| 23 | 10.416060710 | 172.16.14.1 | 172.16.14.63 | ICMP | 98 | Echo (ping) request id=0x0004, seq=3/768, ttl=64 (no respons… |
| 24 | 10.416735151 | 172.16.14.3 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=3/768, ttl=64 |
| 25 | 10.417626492 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=3/768, ttl=64 |
| 26 | 11.417888959 | 172.16.14.1 | 172.16.14.63 | ICMP | 98 | Echo (ping) request id=0x0004, seq=4/1024, ttl=64 (no respon… |
| 27 | 11.418554481 | 172.16.14.3 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=4/1024, ttl=64 |
| 28 | 11.418852347 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=4/1024, ttl=64 |
| 29 | 12.419106357 | 172.16.14.1 | 172.16.14.63 | ICMP | 98 | Echo (ping) request id=0x0004, seq=5/1280, ttl=64 (no respon… |
| 30 | 12.419752012 | 172.16.14.3 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=5/1280, ttl=64 |
| 31 | 12.420049696 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=5/1280, ttl=64 |
| 32 | 13.420301012 | 172.16.14.1 | 172.16.14.63 | ICMP | 98 | Echo (ping) request id=0x0004, seq=6/1536, ttl=64 (no respon… |
| 33 | 13.420937726 | 172.16.14.3 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=6/1536, ttl=64 |
| 34 | 13.421681971 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0004, seq=6/1536, ttl=64 |

Figura 5.1: Finestra Wireshark dell'host H1 nel Test 2.1

arp or icmp

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 5 | 29.035576658 | HewlettP_6a:29:4b | Broadcast | ARP | 42 | Who has 172.16.14.1? Tell 172.16.14.2 |
| 6 | 29.036126495 | HewlettP_15:79:66 | HewlettP_6a:29:4b | ARP | 60 | 172.16.14.1 is at f4:39:09:15:79:66 |
| 7 | 29.036126625 | HewlettP_15:79:be | HewlettP_6a:29:4b | ARP | 60 | 172.16.14.1 is at f4:39:09:15:79:be |
| 8 | 29.036131392 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) request id=0x0001, seq=1/256, ttl=64 (reply in 9) |
| 9 | 29.036703138 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) reply id=0x0001, seq=1/256, ttl=64 (request in 8) |
| 10 | 30.037026390 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) request id=0x0001, seq=2/512, ttl=64 (reply in 11) |
| 11 | 30.038529512 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) reply id=0x0001, seq=2/512, ttl=64 (request in 10) |
| 12 | 31.039067969 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) request id=0x0001, seq=3/768, ttl=64 (reply in 13) |
| 13 | 31.040589038 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) reply id=0x0001, seq=3/768, ttl=64 (request in 12) |
| 14 | 32.041139352 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) request id=0x0001, seq=4/1024, ttl=64 (reply in 15) |
| 15 | 32.042288555 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) reply id=0x0001, seq=4/1024, ttl=64 (request in 14) |
| 16 | 33.042852276 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) request id=0x0001, seq=5/1280, ttl=64 (reply in 17) |
| 17 | 33.044099845 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) reply id=0x0001, seq=5/1280, ttl=64 (request in 16) |
| 18 | 34.044684715 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) request id=0x0001, seq=6/1536, ttl=64 (reply in 19) |
| 19 | 34.046188841 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) reply id=0x0001, seq=6/1536, ttl=64 (request in 18) |
| 20 | 34.289799789 | HewlettP_15:79:66 | HewlettP_6a:29:4b | ARP | 60 | Who has 172.16.14.2? Tell 172.16.14.1 |
| 21 | 34.289833937 | HewlettP_6a:29:4b | HewlettP_15:79:66 | ARP | 42 | 172.16.14.2 is at b0:0c:d1:6a:29:4b |

Figura 5.2: Finestra Wireshark dell'host H2 nel Test 3.1

arp or icmp

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 5 | 17.092881993 | HewlettP_15:79:66 | Broadcast | ARP | 60 | Who has 172.16.14.2? Tell 172.16.14.1 |
| 6 | 17.092941022 | HewlettP_6a:29:4b | HewlettP_15:79:66 | ARP | 42 | 172.16.14.2 is at b0:0c:d1:6a:29:4b |
| 7 | 17.093516192 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0007, seq=1/256, ttl=64 (reply in 8) |
| 8 | 17.093586448 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0007, seq=1/256, ttl=64 (request in 7) |
| 9 | 17.267277813 | HewlettP_15:79:be | Broadcast | ARP | 60 | Who has 172.16.14.2? Tell 172.16.14.1 (duplicate use of 172.16.14.1 detected!) |
| 10 | 17.267333940 | HewlettP_6a:29:4b | HewlettP_15:79:be | ARP | 42 | 172.16.14.2 is at b0:0c:d1:6a:29:4b (duplicate use of 172.16.14.1 detected!) |
| 11 | 17.268771989 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0003, seq=1/256, ttl=64 (reply in 12) |
| 12 | 17.268840079 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0003, seq=1/256, ttl=64 (request in 11) |
| 13 | 18.094838311 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0007, seq=2/512, ttl=64 (reply in 14) |
| 14 | 18.094893512 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0007, seq=2/512, ttl=64 (request in 13) |
| 15 | 18.267607986 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0003, seq=2/512, ttl=64 (reply in 16) |
| 16 | 18.267666045 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0003, seq=2/512, ttl=64 (request in 15) |
| 17 | 19.096562907 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0007, seq=3/768, ttl=64 (reply in 18) |
| 18 | 19.096617261 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0007, seq=3/768, ttl=64 (request in 17) |
| 19 | 19.278850436 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0003, seq=3/768, ttl=64 (reply in 20) |
| 20 | 19.278908018 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0003, seq=3/768, ttl=64 (request in 19) |
| 21 | 20.120530898 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0007, seq=4/1024, ttl=64 (reply in 22) |
| 22 | 20.120587871 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0007, seq=4/1024, ttl=64 (request in 21) |
| 23 | 20.280656901 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0003, seq=4/1024, ttl=64 (reply in 24) |
| 24 | 20.280713628 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0003, seq=4/1024, ttl=64 (request in 23) |
| 25 | 21.144599010 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0007, seq=5/1280, ttl=64 (reply in 26) |
| 26 | 21.144652131 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0007, seq=5/1280, ttl=64 (request in 25) |
| 27 | 21.282028518 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0003, seq=5/1280, ttl=64 (reply in 28) |
| 28 | 21.282077749 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0003, seq=5/1280, ttl=64 (request in 27) |
| 29 | 22.168581397 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0007, seq=6/1536, ttl=64 (reply in 30) |
| 30 | 22.168638886 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0007, seq=6/1536, ttl=64 (request in 29) |
| 31 | 22.282773125 | 172.16.14.1 | 172.16.14.2 | ICMP | 98 | Echo (ping) request id=0x0003, seq=6/1536, ttl=64 (reply in 32) |
| 32 | 22.282819063 | 172.16.14.2 | 172.16.14.1 | ICMP | 98 | Echo (ping) reply id=0x0003, seq=6/1536, ttl=64 (request in 31) |
| 33 | 22.439417675 | HewlettP_6a:29:4b | HewlettP_15:79:be | ARP | 42 | Who has 172.16.14.1? Tell 172.16.14.2 |

Figura 5.3: Finestra Wireshark dell'host H2 nel Test 3.2

arp or icmp

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 23 | 14.749986229 | 172.16.14.1 | 172.16.14.127 | ICMP | 98 | Echo (ping) request id=0x0004, seq=1/256, ttl=64 (no response found!) |
| 24 | 14.750672923 | HewlettP_15:79:66 | Broadcast | ARP | 60 | Who has 172.16.14.1? Tell 172.16.14.127 |
| 25 | 15.751654611 | 172.16.14.1 | 172.16.14.127 | ICMP | 98 | Echo (ping) request id=0x0004, seq=2/512, ttl=64 (no response found!) |
| 26 | 15.770811258 | HewlettP_15:79:66 | Broadcast | ARP | 60 | Who has 172.16.14.1? Tell 172.16.14.127 |
| 28 | 16.775644946 | 172.16.14.1 | 172.16.14.127 | ICMP | 98 | Echo (ping) request id=0x0004, seq=3/768, ttl=64 (no response found!) |
| 29 | 16.794759912 | HewlettP_15:79:66 | Broadcast | ARP | 60 | Who has 172.16.14.1? Tell 172.16.14.127 |
| 31 | 17.799628817 | 172.16.14.1 | 172.16.14.127 | ICMP | 98 | Echo (ping) request id=0x0004, seq=4/1024, ttl=64 (no response found!) |

Figura 5.4: Finestra Wireshark dell'host H2 nel Test 5.2

Ben fatta e approfondita

11/10