Escucha de paquetes ARP y análisis de la entropía del sistema: conclusiones acerca de la importancia de cada nodo

Alejandro Danós, Claudio Graiño y Luis Scoccola

***Resumen*—Basándonos en la escucha de mensajes ARP analizamos la estructura de una red local. Es decir, qué nodos se comportan como *host* y cuáles como *server*. Modelamos el flujo de mensajes como una fuente de información y calculamos la entropía de diversas maneras. Nos preguntamos cuán significativa es la información proporcionada por los modelos estudiando el protocolo y sus implementaciones.**

***Index Terms*—ARP, LAN, server, host, entropía, información, Ethernet, Wi-Fi**

I. Introduccíon teórica

PARA estudiar el protocolo ARP (*Address Resolution Protocol*) nos proponemos analizar la estructura de una red local capturando y analizando este tipo de mensajes.

El protocolo se encuentra entre las capa de red y la capa de enlace (capas 3 y 2 respectivamente). Cumple una función crucial en redes de acceso múltiple ya que es el encargado de traducir direcciones de red en direcciones de enlace.

Los mensajes tienen una estrucura muy simple[2, Packet format]:

```
mensaje ARP {
      especificaciones de protocolos
      tipo
      direccion red fuente
      direccion red destino
      direccion enlace fuente
      direccion enlace destino
}
```

`especificaciones de protocolos` se usa para *parsear* el paquete y especificar los protocolos de nivel 3 y 2 que deben comunicarse, y `tipo` puede tomar dos valores, interpretados de la siguiente forma:

- *who-has* El remitente necesita saber la dirección de enlace correspondiente a un IP dado.
- *is-at* Se está dando a conocer la dirección de enlace correspondiente al IP del remitente.

Usando estos campos buscamos deducir la mayor cantidad de información de una red local dada.

Para esto modelamos el flujo de mensajes ARP como una fuente de información. Probamos varias elecciones de símbolos para encontrar la más adecuada a nuestros fines.

II. Desarrollo

COMO se estipula en [1], implementamos algunos programas que nos permitan observar el funcionamiento del protocolo y condensar la información recopilada.

Para esto usamos Python junto con Scapy[6], una librería dedicada al manejo de mensajes y paquetes de varios protocolos.

A. Ejercicio 1

El primer programa[1] nos permite enviar paquetes ARP con parámetros arbitrarios y recibir respuestas en el caso en que las haya.

Analizamos el comportamiento del protocolo ante pedidos particulares y casos patológicos, buscando una correspondencia con las especificaciones de [2] y [4].

B. Ejercicio 2

Una vez entendido el mecanismo básico del protocolo, implementamos un monitor de paquetes. Usamos ARP en el contexto IPv4/Ethernet o IPv4/Wi-Fi. El monitor simplemente escucha cada mensaje que es transmitido por el medio compartido y, si es un paquete ARP, lo *parseamos* y guardamos convenientemente [2].

Llegado este momento nos proponemos identificar al router que se comporta como el *gateway* por defecto de una red local. Para esto buscamos la dirección IP cuya probabilidad de aparecer como remitente o destinatario de un mensaje de tipo *who-has* sea la más alta.

Es decir, tomando como fuente de información la direccíon IP de los remitentes de los mensajes ARP de tipo *who-has*, buscaremos el símbolo cuya información sea la menor. Hacemos algo análogo para los destinatarios de dichos mensajes. Además de ser algo intuitivo, esta idea se basa en el comentario final de [2, Related issue]. En pocas palabras, sugiere que las tablas internas de los monitores ARP de cada interfaz contengan un *time-out* por cada entrada. El objetivo es minimizar la cantidad de información inconsistente con la realidad. Esta práctica parece particularmente acertada cuando se usa el protocolo 802.11 o semejantes, donde la probabilidad de que un nodo de la red local se desconecte resulta bastante alta [3].

B.1 Entropía de la red

Además del modelo de fuente de información basado en los remitentes de los paquetes de tipo *who-has* conside-

A. Danós, e-mail: adp007@gmail.com
C. Graiño, e-mail: claudio.cuervo.17@gmail.com
L. Scoccola, e-mail: luis.scoccola@gmail.com

[1] Ver archivo `ARPclient.py`.
[2] Ver archivo `ARPmoitor.py`.
[3] Ya sea porque se perdió conectividad, porque se eligió a una red por sobre otra o porque se trata de una red local pública donde una IP puede estar asociada a cientos de direcciones MAC distintas en un mísmo día.

ramos otras fuentes de información[4]. Para cada instancia analizaremos la entropía de seis maneras distintas, tomando como símbolos:

1. *IP origen - IP destino* mensajes de tipo *who-has*
2. *IP origen - IP destino* mensajes de tipo *is-at*
3. *IP origen* mensajes de tipo *who-has*
4. *IP origen* mensajes de tipo *is-at*
5. *IP destino* mensajes de tipo *who-has*
6. *IP destino* mensajes de tipo *is-at*

B.2 Información de cada nodo

Para cada uno de estos modelos calculamos la cantidad de información que conlleva cada evento. La idea es analizar qué nodos se encuentran por debajo y por arriba de la entropía y así distiguir a los más activos.

Suponemos que el o los *gateways* de la red local serán los nodos cuya información sea la menor. Para corroborarlo analizaremos primero instancias con topología y direcciones conocidas.

REALIZAMOS este análisis en varias redes locales. La idea es estudiar en qué casos la hipótesis de que el router se corresponde con el nodo de menor información, se verifica.

III. Resultados

A. Ejercicio 1

DIVIDIMOS los resultados de los experimentos de acuerdo a qué características del protocolo involucran.

A.1 Algoritmo de respuesta

Comenzamos realizando un pedido por una IP de la red local que se encuentre asignada a una interfaz encendida. Comprobamos que responde instantáneamente con los campos especificados en [2, Packet Reception]. Es decir, intercambiando direcciones MAC y direcciones IP y escribiendo su propia dirección MAC en el campo de dirección fisica del remitente.

Al realizar un pedido por una IP que no se encuentra asignada a una interfaz encendida no recibimos respuesta alguna, como es de esperarse.

Cuando mandamos un mensaje ARP de tipo *who-has* con la propia IP del remitente como destino tampoco recibimos respuesta.

Lo mismo ocurre al usar como destino la IP de la red o la IP de *broadcast*.

A.2 Algoritmo de resolución de conflictos

Luego realizamos un pedido que cree un conflicto del tipo descripto en [3, 2.4. Ongoing Address Conflict Detection and Address Defense]. Para esto enviamos paquetes de tipo *who-has* con una IP de fuente igual al de alguna interfaz encendida y funcionando en el momento. El pedido es a una IP cualquiera, dado que no forma parte de lo contemplado por el experimento. Esperamos capturar la reacción de algunas de las interfaces que interactúan en la red local. En particular, la de la interfaz con la IP en cuestión y la del router. Para esto utilizamos el monitor de paquetes ARP. Aquí la dirección MAC `74:e5:0b:16:82:5e` se corresponde con la de la interfaz que crea el conflicto. La misma se adjudica la IP `192.168.0.5` que se encuentra asociada a la direccón MAC `00:13:e8:cd:eb:41` en las tablas del router y de la interfaz en cuestión.

```
------MACsrc----- ----IPsrc-- ----IPdst---

74:e5:0b:16:82:5e 192.168.0.5 192.168.0.9 <- conf
3c:75:4a:f2:b8:c3 192.168.0.1 192.168.0.5 <- router
00:13:e8:cd:eb:41 0.0.0.0     192.168.0.5 <- test
00:13:e8:cd:eb:41 192.168.0.5 192.168.0.1
00:13:e8:cd:eb:41 0.0.0.0     192.168.0.5 <- test
00:13:e8:cd:eb:41 192.168.0.5 192.168.0.1
00:13:e8:cd:eb:41 0.0.0.0     192.168.0.5 <- test
00:13:e8:cd:eb:41 192.168.0.5 192.168.0.1
...               ...         ...
```

B. Ejercicio 2

B.1 Primera instancia: red principalmente *wireless*

Realizamos una captura de tres horas[5] de mensajes transmitidos por una red local hogareña.

Sabemos que la red constaba, entre celulares y computadoras, de a lo sumo nueve nodos reales. Es decir, nueve interfaces. De las seis fuentes de información analizadas encontramos que la más entrópica es la que toma como símbolos a la dupla *IP orígen - IP destino* de los mensajes de tipo *who-has*.

Observamos que, como conjeturamos, la menor cantidad de información está dada por la IP asignada al router, tanto para los orígenes como para los destinos de los mensajes de tipo *who-has*.

Además notamos que varias IPs no se corresponden con IPs locales. Para entender esta situación realizamos el análisis explicado en **Ejercicio 3**.

Entropías calculadas:

```
1)    2.78012618598
2)    0.642857913091
3)    0.464354772895
4)    0.485029674026
5)    0.485009539485
6)    0.485029674026
```

B.2 Segunda instancia: red principalmente cableada

Realizamos una captura de 40 minutos[6] en otra red local hogareña.

En este caso también vemos que el modelo más entrópico es el (1). Pero nos llevamos una sorpresa al notar que la fuente de menor informacíon en los modelos (3) y (5) no se corresponde con el router.

Entropías calculadas:

```
1)    2.4260538216
2)    0.266384463268
3)    0.160202479202
4)    0.149786613678
5)    0.321879422676
6)    0.184443972706
```

[4] Ver archivo `plotiere.py`.

[5] Captura en archivo `casa_21_16_3h.dat`.

[6] Captura en archivo `captura-40-minutos-domingo-16-55.dat`.

B.3 Tercera instancia: descubriendo nodos mediante ARP

Corremos un programa que, dado un rango de IPs, envía un mensaje de tipo *who-has* a cada uno[7].

Entropías calculadas:

```
1)    3.19634702574
2)    1.04813198514
3)    0.218762854699
4)    0.0838822430187
5)    0.584411124474
6)    0.0838822430187
```

C. Instancias propocionadas por la cátedra

Para probar nuestro modelo analizamos dos instancias tomadas de una red con nodos desconocidos.

C.1 Instancia Pequeña

Entropías calculadas:

```
1)    4.31636529084
2)    0.0886346203114
3)    0.35590656708
4)    0.0383736259199
5)    0.364981009758
6)    0.036805419629
```

C.2 Instancia Grande

Entropías calculadas:

```
1)    4.89169346855
2)    0.232146795312
3)    0.295031555575
4)    0.0276246997642
5)    0.291162905732
6)    0.026694835945
```

D. Ejercicio 3

Dada una instancia con cualquier número de capturas construimos dos grafos dirigidos. Uno representa al flujo de mensajes de tipo *who-has* y el otro a los de tipo *is-at*. La interpretación de los mismos se resume:

NODOS Cada nodo representa una IP. El tamaño de un nodo depende de la probabilidad de ser remitente o destinatario de los paquetes. Es decir, es inversamente proporcional a la cantidad de información que proporciona el evento.

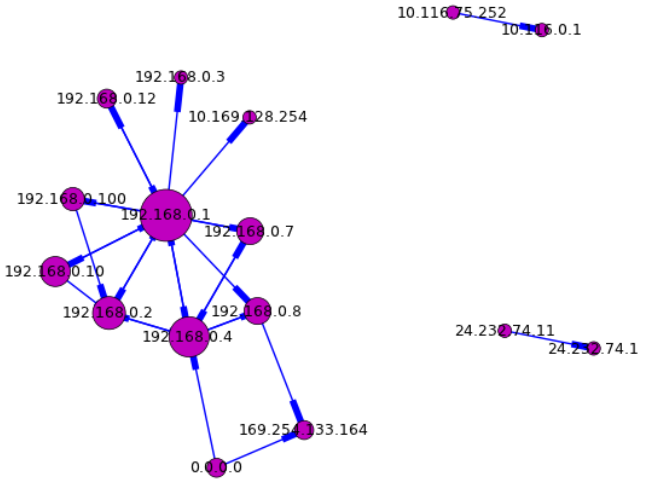
EJES Un nodo n_1 presenta un eje dirigido hacia otro nodo n_2 *sii* hemos capturado un paquete con remitente n_1 y destinatario n_2.

Cuando las redes contienen muchos nodos, los grafos son poco significativos, pues es difícil distiguirlos entre ellos visualmente.

Para este tipo de instancias graficamos la cantidad de información en función de la IP, tanto para IP fuente como para IP destino.

[7] Ver archivo `ARPclient-rastrillo.py`.

D.1 Primera instancia



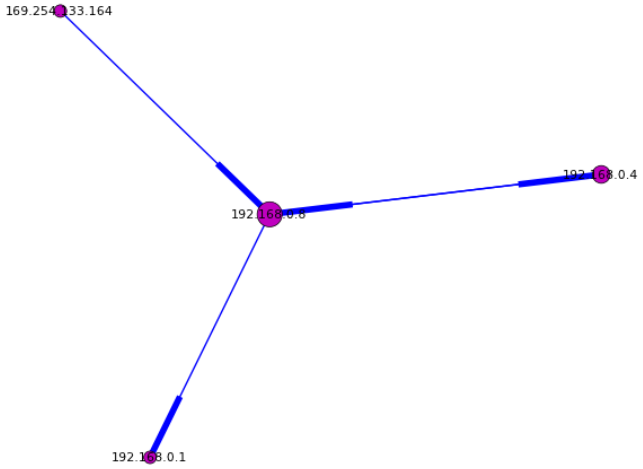
Mensajes *who-has* en red *wireless*

El grafo se compone de tres componentes conexas.

1. Observamos que la componente `10.116.75.252 - 10.116.0.1` parece estar generada por mensajes enviados por el router. Esto lo deducimos de la dirección MAC fuente de dichos mensajes. Basándonos en más análisis de la misma red vemos que estos mensajes son enviados cada 22 ± 1 minutos. Analizando el tráfico con TCPDUMP [7] no encontramos más paquetes con el destino y el remitente mencionados.
2. La componente `24.232.74.11 - 24.232.74.1` tiene como remitente a la IP pública del router y, como dirección MAC, la dirección de la interfaz que da al *exterior* de dicho router. Estos paquetes son enviados cada media o una hora.
3. En la componente principal tenemos en el centro al router rodeado de IPs que se corresponden con ocho de las nueve interfaces pertenecientes a la red. Notamos que el router pregunta por una IP privada pero que no pertenece al dominio `192.168.0.0/24`. No encontramos más tráfico de este tipo, ni pudimos asociarlo a alguno de los comportamientos mencionados en las referencias.

   En esta misma componente notamos la IP `0.0.0.0` y la `169.254.133.164`. Los mensajes llevan como remitente la dirección MAC de una de las interfaces de la red.

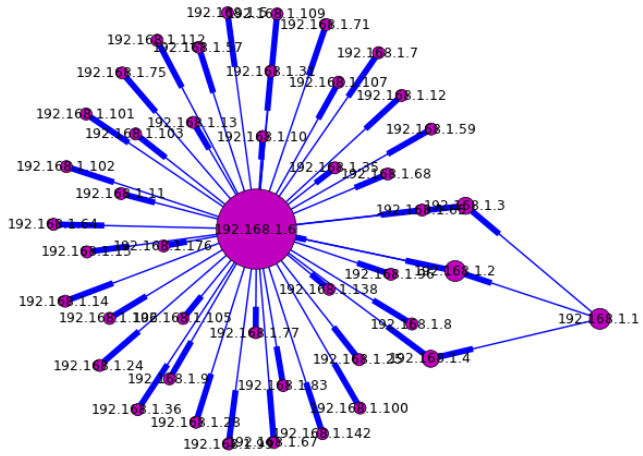
El grafo de respuestas (*is-at*) presenta muchos menos nodos. Además no es el router el nodo central, sino uno de los *hosts*.



Mensajes *is-at* en red *wireless*

D.2 Segunda instancia

El nodo central, el que presenta mayor actividad, no es el router en este caso. Notamos la gran cantidad de IPs por los que se pregunta, cuando la red consta de a lo sumo cuatro nodos (entre computadoras y celulares). Además el router, la IP `192.168.1.1`, no parece preguntar nunca por el nodo central.

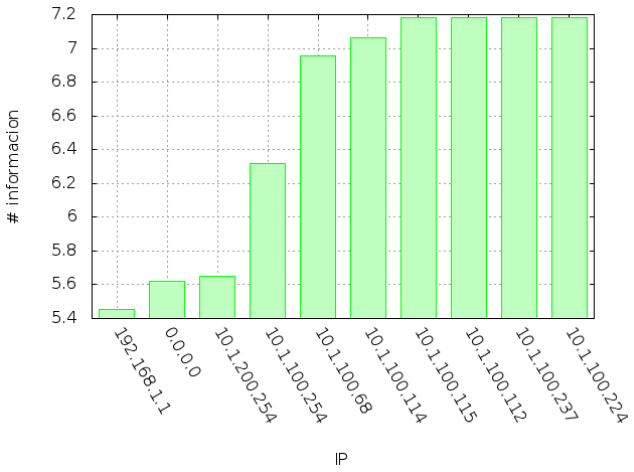


Mensajes *who-has* en red cableada

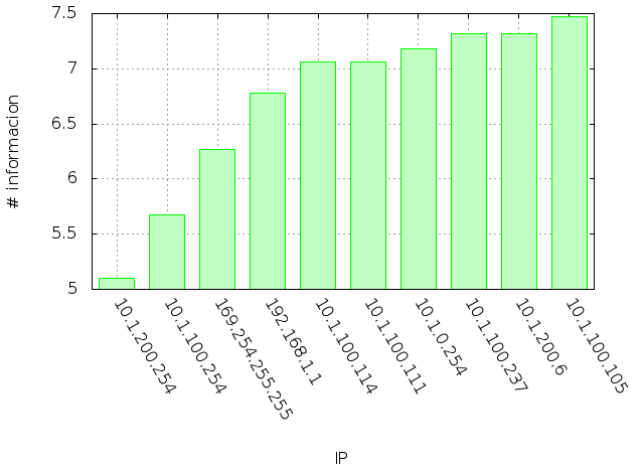
E. Instancias proporcionadas por la cátedra

Ya que ambas instancias presentaban una gran cantidad de nodos, graficamos la información en función de la IP de los diez nodos que presentan la menor cantidad de información, tanto para IP fuente como destino.

E.1 Instancia Pequeña

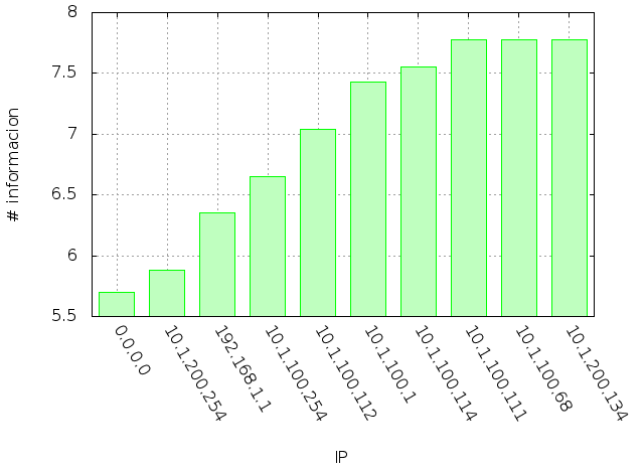


Cantidad de información - IP fuente

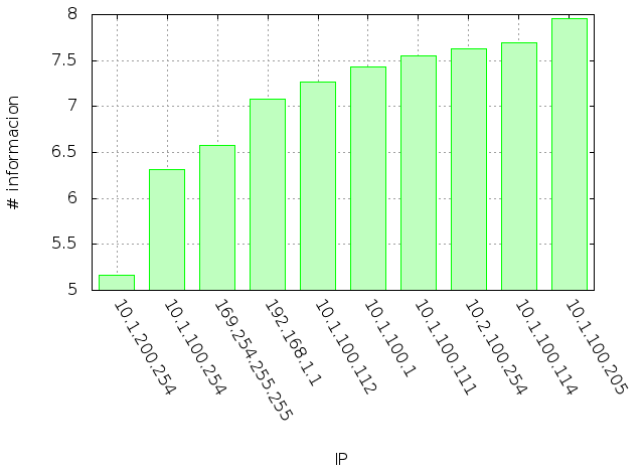


Cantidad de información - IP destino

E.2 Instancia Grande



Cantidad de información - IP fuente



Cantidad de información - IP destino

IV. Discusión

AQUÍ presentamos nuestras deducciones acerca de los resultados expuestos anteriormente.

A. Ejercicio 1

A.1 Algoritmo de respuesta

Todo mensaje con destinatario una IP que no se corresponde con una interfaz en funcionamiento no es respondido, simplemente porque nadie se ve identificado con dicha IP. Éste también es el caso de la IP de la red local.

Sin embargo, la razón por la cual un mensaje de tipo *who-has*, con destino igual al de la IP asignada al remitente, no es respondido por esa misma interfaz no se encuentra explicada en [2, Packet Reception] ya que el algoritmo propuesto sólo chequea la guarda `?Am I the target protocol address?`.

Este estándar es bastante antiguo y el protocolo implementado por las interfaces modernas tiene varias optimizaciones. Una de ellas es el uso de mensajes ARP con el objetivo de saber si una IP dada está libre o asignada a una interfaz de la red local.

Este tipo de mensajes justamente tiene como destinatario la IP en cuestión. A continuación se analiza con más detalle este tipo de paquetes.

A.2 Algoritmo de resolución de conflictos

En una primera instancia el router, al recibir información contradictoria, pregunta por la dirección MAC asociada a la IP en conflicto.

Por otro lado suponemos que la interfaz deja de usar esa IP momentáneamente para realizar el test estipulado en [3, 2.1.1. Probe Details], donde se especifica que la IP del remitente y la dirección MAC del destinatario de estos paquetes deben ser 0.

Los mensajes destinados al router no se encuentran explicados en el estándar citado pero conjeturamos que son una forma de reclamarle al router el IP en conflicto. Además no logramos reproducir el comportamiento indicado en [4, 2.3. Announcing an Address]. Estimamos que puede deberse a que es un estándar del 2008 y puede no estar implementado en la interfaz en cuestión.

B. Ejercicio 2

B.1 Primera instancia

1. No logramos relacionar este comportamiento con ninguno de los especificados en las referencias.
2. Suponemos que se trata de una comunicación entre el router (el *gateway* de la red local) y el *gateway* de la red a la que pertenece la interfaz que mira al *exterior* del router. Esta suposición está sustentada por la dirección de IP del destinatario: la primera del conjunto de direcciones posibles. Conjeturamos que estos mensajes son enviados a la red local por un error de programación en el router. Si bien, analizando el tráfico con TCPDUMP, no encontramos evidencia de que otros paquetes sean filtrados a la red local.
3. Este comportamiento es explicado en [4, 2. Address Selection, Defense and Delivery]. Básicamente, cuando una interfaz pierde conectividad, o es encendida, en el caso de no recibir una IP mediante DHCP se asigna una de forma aleatoria en el rango `169.254.1.0` – `169.254.254.255`. Luego realiza un proceso análogo al que capturamos al generar un conflicto de IPs. Este se encuentra descripto en [4, 2.4. Announcing an Address].

Las respuestas, como se recomienda en [2, An Example] son unicast. Es por esta razón que no fueron capturadas. No pudimos extraer información significativa de estos paquetes. En lo que sigue trataremos sobre todo lo mensajes *who-has*.

B.2 Segunda instancia

Dado que un mensaje de tipo $who-has$ sólo puede ser respondido por la interfaz que tiene asignado la IP en cuestión, cada mensaje de este tipo es habitualmente transmitido por toda la red local ($flooding$). Por este motivo, aún en una red *switcheada*, recibiremos estos mensajes.

El nodo con menor cantidad de información es uno de los *hosts*. Estimamos que está corriendo algún algoritmo de detección de *hosts*. No nos queda claro el por qué de este algoritmo pero puede deberse a una elección de implementación de la interfaz o a *malware* entre otros.

A pesar de la razón de este comportamiento, nos ayuda a explicar por qué el router jamás pregunta por la interfaz en cuestión.

Como ya mencionamos, en [2], se sugiere que cada entrada de la tabla interna de cada interfaz contenga un *time-out*. Como los mensajes de tipo *who-has* son enviados mediante *broadcast* cada interfaz que reciba uno renovará la entrada correspondiente en su tabla. Como el nodo `192.168.1.6` envía constantemente paquetes de tipo *who-has* suponemos que el router renueva constantemente el *time-out* de su tabla, y es por esto que nunca tiene la necesidad de interactuar con el *host*.

No nos queda claro por qué el *host* jamás pregunta por el router.

EN ambos casos el modelo más entrópico resultó ser el (1). Esto es entendible mirando los gráficos: la mayor cantidad de intercambios, que podemos recibir, se da con mensajes *who-has*. Y tomando como símbolos las duplas tenemos un mayor conjunto de símbolos posibles en comparación con los modelos en donde los símbolos son una única IP.

B.3 Tercera instancia

Esto mismo se verifica en la instancia 3, que podríamos denominar patológica. Basándonos en lo observado en la segunda instancia corrimos, en la misma red local, un algoritmo similar para descubrir nodos. Notamos que la entropía del modelo (1) es aún más alta, aunque la cantidad de nodos sigue siendo la misma. Por otro lado, vemos que los mensajes *it-at* parecen brindarnos aún menos información.

C. Instancias propocionadas por la cátedra

Primero notamos que, en ambos casos, la entropía del modelo de tipo (1) es la más alta, como venimos constatando a lo largo del trabajo. La entropía de la instancia grande es mayor a la de la instancia pequeña. De aquí deducimos que hay un mayor intercambio de mensajes. Es decir, existen más pares *IP fuente - IP destino*.

Además los resultados de los seis modelos de la instancia grande se asemejan a los de la pequeña. Si los comparamos con las otras entropías calculadas, en las otras instancias, no encontraremos esta similitud. Con lo cual suponemos que se trata de la misma red y que la diferencia reside en el momento y en la cantidad de tiempo que se realizó la captura.

Esta hipótesis es constatada al ver los gráficos de Informacion vs IP. Vemos que, tanto para las IP fuente como las destino, el conjunto de las primeras parece ser el mismo en ambas instancias.

En el siguiente analisis tomamos en cuenta ambas instancias como dos capturas de la misma red.

La IP `10.1.200.245` se encuentra primera como destino y segunda como fuente en la instancia grande (y figura entre las primeras en la instacia pequeña). Esto parece corroborar la hipótesis de que una información más baja se corresponderá con el router, dado que es una dirección que suele asignarse a routers (es la última dirección del rango). Esta misma observación aplica a la IP `10.1.200.245` y la IP `192.168.1.1` que también figuran en ambas instancias. Con lo cual deducimos que la red constaba de más de un router o switch[8].

Otras direcciones que observamos son:

- `0.0.0.0` como IP fuente: Ya vimos que es usada por cualquier *hosts* para chequear que una IP no esté siendo usada.
- `196.254.255.255` como IP destino: Se trata de la dirección de *broadcast* del *link-local*. En ninguno de los estándares encontramos mención de mensajes ARP con esta dirección IP como destino. Sin embargo en [5] se propone este tipo de mensajes como parte de un método para detectar defectos en un router. Suponemos que estamos observando la implementación de este método, si bien nos llama la atención la gran cantidad de mensajes con este fin.

[8] No podremos diferenciar entre routers o switchs realizando este anáisis.

V. Conclusiones

VIMOS que, basándonos únicamente en paquetes ARP, la mayor cantidad de información de la red se obtiene analizando los mensajes de tipo *who-has*. Esto se corresponde con la entropía medida, que es justamente la esperanza de la cantidad de informacíon dada, en este caso, por cada nodo.

Si cada interfaz implementa el protocolo básico, será posible diferenciar entre nodos activos y menos activos, deduciendo cuáles se comportan como *server* y cuáles como *host*. Pero esta interpretación es muy susceptible a comportamientos no especificados en el estándar. Sobre todo en redes con pocos nodos. A modo de ejemplo, un nodo que esté descubriendo nodos mediante mensajes ARP puede entorpecer el estudio. Este tipo de comportamiento podrá producir una entropía muy elevada en una red que consta de pocos nodos, y por lo tanto las deducciones podrían ser poco significativas, como mostramos en el análisis de la tercera instancia en **Discusión**.

Para instancias con muchos nodos mostramos que el análisis de la función de los nodos (*server* - *host*) basado en paquetes ARP puede resultar efectivo sin necesidad de capturar más paquetes que los que son enviados mediante *broadcast*. Es decir, sin realizar ataque alguno, logramos diferenciar al *server*.

Referencias

[1] Cátedra de Teoría de las Comunicaciones
Primer trabajo práctico
Primer cuatrimestre 2013
[2] RFC 826 - Ethernet Address Resolution Protocol
`http://tools.ietf.org/html/rfc826`
C. Plummer 1982
[3] RFC 5227 - IPv4 Address Conflict Detection
`http://tools.ietf.org/html/rfc3927`
S. Cheshire 2008
[4] RFC 3927 - Configuration of IPv4 Link-Local Addresses
`http://tools.ietf.org/html/rfc3927`
Cheshire, et al. 2005
[5] Method and apparatus for detecting a router that improperly responds to ARP requests
US 7729292 B2
`http://www.google.com/patents/US7729292`
Stuart D. Cheshire y Joshua V. Graessley
[6] `http://www.secdev.org/projects/scapy`
[7] `http://www.tcpdump.org`