Brno University of Technology
Faculty of Information Technology

Computer Communications and Networks

ARP Scanner

April 23, 2017

Svätopluk Hanzel

Contents

1 Project assignment

The main task of this project was to create a network tool capable of scanning local networks using *Address Resolution Protocol* (ARP). Result of such a scan should reveal MAC addresses of computers on network and their corresponding IP addresses.

2 Address resolution protocol

ARP is a critical network protocol for IPv4 networks. It is used for mapping network address to physical address [1][2].

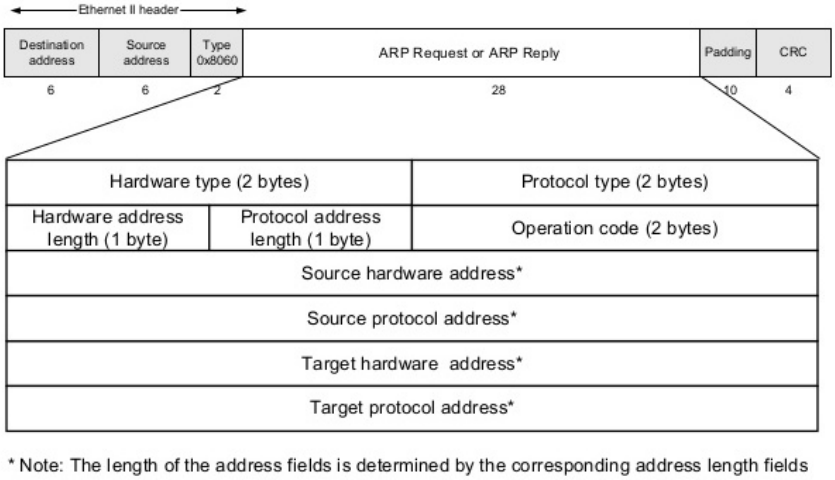

Figure 1: ARP packet format

Source: WikiMedia

2.1 ARP request

Whenever computer A (IP address 10.0.0.42) wants to communicate with computer B (IP address 10.0.0.69), it first has to know the B's physical (MAC) address. First, A looks into its *ARP cache*, where it has a list of known MAC addresses and corresponding IP addresses [3]. If no record was found for B's IP address, A has to issue an *ARP request* before any further requests.

However, since A doesn't know B's physical address, this request has to be sent to a *BROADCAST* address, which means that the L2 header's destination field contains `FF:FF:FF:FF:FF:FF`. Using this approach, all computers in the network receive the ARP request [4].

2.2 ARP reply

Whenever host receives ARP request, it checks whether his IP address matches the one specified in the ARP request. If it does, its ARP cache is updated with the address mapping of the sender of the ARP request. The host then creates an ARP reply message which contains the requested MAC address and is sent using unicast directly to the sender of the ARP request [4].

3 Project solution

Solution of this project was rather straight-forward. It takes the following steps:

1. Parse arguments
2. Gather interface information
3. Send ARP requests and listen for replies
4. Send ICMPv6 echo request for IPv6 scanning
5. Generate XML file

3.1 Argument parsing

The program takes 2 mandatory arguments: `-i`, which specifies the interface to scan with and `-f` to specify the XML output file. The argument parsing uses the standard `getopt` function.

3.2 Interface information gathering

Before any requests can be made, interface informations have to be gathered. This is done in using the `Interface` class, which discovers all informations about the specified interface including its index, MAC address, all IPv4 addresses and corresponding netmasks and all IPv6 link-local addresses.

All these informations are printed out for debugging/informative reasons.

3.3 ARP requests processing

After all the informations about the specified interface are available, 2 new RAW sockets are opened. One for listening for ARP replies and the other one for sending ARP requests.

A new thread is created for receiving ARP replies. This allows us to receive ARP replies even while other ARP requests are still being sent. In this thread the ARP packets are being received until `Ctrl+C` is pressed (or `SIGINT` is sent). Each new host's IP and MAC addresses are then saved for later XML generation.

ARP requests on the other hand are sent in the main thread. ARP requests are sent to all IP addresses in subnet of each address on the interface, i.e. if the interface has 2 IPv4 addresses (`10.0.0.2/24` and `10.55.0.3/24`) ARP request is sent to each IP address in the `10.0.0.0/24` and the `10.55.0.0/24` subnet.

3.4 IPv6 scanning

Since IPv6 networks are too big to scan, we cannot simply iterate through all the IPs and send them some request. IPv6 also doesn't have ARP or broadcast. However, in case we need to get all IPv6-enabled hosts in our network we may send a *ICMPv6 echo request* to solicited node address `ff02::1` to which all devices on our local network belong. After that, all these devices should respond with echo reply [5].

In this solution, this technique isn't fully implemented, since Windows machines do not respond to valid multicasted ICMPv6 echo requests and this program sends only valid echo requests.

3.5 XML generation

Once the program received `SIGINT` signal, it attempts to finish all threads as soon as possible. Once this is done, XML file content is generated and written out to the file.

References

[1] Plummer, D. C. *Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware* STD 37 RFC Editor November 1982 URL `http://www.rfc-editor.org/rfc/rfc826.txt` `http://www.rfc-editor.org/rfc/rfc826.txt`.

[2] Wikipedia *Address Resolution Protocol* online URL `https://en.wikipedia.org/wiki/Address_Resolution_Protocol`.

[3] Corporation, M. *ARP Cache* online URL `https://technet.microsoft.com/en-us/library/cc958841(en-us).aspx`.

[4] Corporation, M. *ARP Process* online URL `https://technet.microsoft.com/en-us/library/cc940010.aspx`.

[5] Gont, F.; Chown, T. *Network Reconnaissance in IPv6 Networks* RFC 7707 RFC Editor March 2016.