# Exp#6a Simulation of ARP Protocols

Date:

Aim

To know the address of a host when its logical address is known using ARP

protocol.

Algorithm

Target/Server

1. Create a server socket.
2. Accept client connection.
3. Read IPaddress from the client request
4. Check its configuration file and compare with its logical address.
5. If there is a match,send the host physical address.
6. Stop

Client

1. Create a socket.
2. Send IP address to the target machine
3. Receive target's response
4. If It is a MAC address then display it and goto step 6
5. Display"Host not found"
6. Stop

Program

// ARP Server –ArpServer.java

**import** java.io.\*;

**import** java.net.\*;

**public** **class** ArpServer {

**public** **static** **void** main(String[] args) {

**try** (ServerSocket serverSocket = **new** ServerSocket(2500)) {

System.***out***.println("ARP Server started on port 2500...");

**try** (Socket clientSocket = serverSocket.accept()) {

System.***out***.println("Client connected.");

BufferedReader br = **new** BufferedReader(**new** InputStreamReader(clientSocket.getInputStream()));

PrintStream ps = **new** PrintStream(clientSocket.getOutputStream());

String requestedIP = br.readLine().trim();

System.***out***.println("Requested IP: " + requestedIP);

String macAddress = *getMacFromIpconfig*(requestedIP);

**if** (macAddress != **null**) {

ps.println("MAC Address: " + macAddress);

} **else** {

ps.println("IP address not found or MAC address not available.");

}

}

} **catch** (IOException e) {

System.***err***.println("Server Error: " + e);

}

}

// Helper method to run ipconfig and extract MAC address

**private** **static** String getMacFromIpconfig(String ipAddress) {

**try** {

Process process = Runtime.*getRuntime*().exec("ipconfig /all");

BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(process.getInputStream()));

String line;

**boolean** foundIPBlock = **false**;

String macAddress = **null**;

**while** ((line = reader.readLine()) != **null**) {

line = line.trim();

// If we're in the relevant block and find "Physical Address", capture it

**if** (line.toLowerCase().startsWith("physical address")) {

**int** index = line.indexOf(":");

**if** (index != -1) {

macAddress = line.substring(index + 1).trim(); }

}

// Look for the requested IP address

**if** (line.contains(ipAddress)) {

foundIPBlock = **true**;

}

// Reset if a new interface block starts (indicated by an empty line)

**if** (line.isEmpty()) {

foundIPBlock = **false**;

}

}

reader.close();

**if**(foundIPBlock){

**return** macAddress;}**else**{

**return** macAddress = **null**;

}

} **catch** (IOException e) {

System.***err***.println("Failed to execute ipconfig: " + e);

**return** **null**;

}

}

}

// ARP Client -- ArpClient.java

import java.io.\*;

import java.net.\*;

public class ArpClient {

public static void main(String[] args) {

try {

// Connect to ARP server on localhost and port 2500

Socket client = new Socket("localhost", 2500);

System.out.println("Connected to ARP Server.");

// Input from user

BufferedReader userInput = new BufferedReader(new InputStreamReader(System.in));

// Output stream to server

PrintStream out = new PrintStream(client.getOutputStream());

// Input stream from server

BufferedReader in = new BufferedReader(new InputStreamReader(client.getInputStream()));

// Prompt user for IP address

System.out.print("Enter the IP address: ");

String ipAddress = userInput.readLine();

// Send IP address to server

out.println(ipAddress);

// Receive MAC address or error from server

String response = in.readLine();

// Display the response

if (response == null || response.trim().isEmpty()) {

System.out.println("Host does not exist or MAC address not found.");

} else {

System.out.println("Response from Server: " + response);

}

// Close all connections

in.close();

out.close();

client.close();

userInput.close();

} catch (IOException e) {

System.err.println("Client Error: " + e.getMessage());

}

}

}

**Output:**

Server

$ javac ArpServer.java

$ java ArpServer Server started

Client

$ javac ArpClient.java

$ java ArpClient

Enter the IP address : 172.16.12.251 Physical Address B8:AC:6F:1B:AB:06

Result

Thus using ARP protocol,server’sMAC address is obtained

# Exp#6b Simulation of RARP Protocols

Date:

Aim

To know the logical address of a host when its physical address is known using RARP

protocol.

Algorithm

Target/Server

1. Create a server socket.
2. Accept client connection.
3. Read MAC address from the client request
4. Check its configuration file and compare with its physical address.
5. If there is a match,send the host logical address.
6. Stop

Client

1. Create a socket.
2. Send physical address to the target machine
3. Receive target's response
4. If it is a Ip address then display it and goto step6
5. Display"Host not found"
6. Stop

**Program**

//RARPServer--RarpServer.java

**package** sakthivel;

**import** java.io.\*;

**import** java.net.\*;

**public** **class** RarpServer {

**public** **static** **void** main(String[] args) {

**try** (ServerSocket serverSocket = **new** ServerSocket(2500)) {

System.***out***.println("RARP Server started. Waiting for client...");

**try** (Socket clientSocket = serverSocket.accept()) {

System.***out***.println("Client connected.");

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(clientSocket.getInputStream()));

PrintStream out = **new** PrintStream(clientSocket.getOutputStream());

String macInput = in.readLine().trim().toLowerCase().replace("-", ":"); // Normalize MAC

System.***out***.println("Received MAC address: " + macInput);

// Run ipconfig /all

Process process = Runtime.*getRuntime*().exec("ipconfig /all");

BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(process.getInputStream()));

String line;

**boolean** macMatched = **false**;

String foundIP = **null**;

**while** ((line = reader.readLine()) != **null**) {

line = line.trim();

// Extract MAC address first

**if** (line.toLowerCase().startsWith("physical address")) {

**int** idx = line.indexOf(":");

**if** (idx != -1) {

String mac = line.substring(idx + 1).trim().toLowerCase().replace("-", ":");

**if** (mac.equals(macInput)) {

macMatched = **true**;

} **else** {

macMatched = **false**;

}

}

}

// If MAC matched, look for IP address next

**if** (macMatched && (line.toLowerCase().startsWith("ipv4 address") || line.toLowerCase().contains("ip address"))) {

**int** idx = line.indexOf(":");

**if** (idx != -1) {

foundIP = line.substring(idx + 1).trim().replace("(Preferred)", "").trim();

**break**;

}

}

}

reader.close();

**if** (foundIP != **null**) {

out.println("IP Address: " + foundIP);

} **else** {

out.println("IP address not found for given MAC.");

}

}

} **catch** (IOException e) {

System.***err***.println("RARP Server Error: " + e.getMessage());

}

}

}

//RARPClient--RarpClient.java

import java.io.\*;

import java.net.\*;

public class RarpClient {

public static void main(String[] args) {

try {

// Allow optional server IP as command-line argument

String serverIP = (args.length > 0) ? args[0] : "localhost";

// Connect to the RARP server

Socket client = new Socket(serverIP, 2500);

System.out.println("Connected to RARP Server at " + serverIP);

// Reader for user input (keyboard)

BufferedReader userInput = new BufferedReader(new InputStreamReader(System.in));

// Output stream to server

PrintStream out = new PrintStream(client.getOutputStream());

// Input stream from server

BufferedReader in = new BufferedReader(new InputStreamReader(client.getInputStream()));

// Prompt user to enter MAC address

System.out.print("Enter the MAC address (e.g., 00-1A-2B-3C-4D-5E or 00:1A:2B:3C:4D:5E): ");

String macAddress = userInput.readLine().trim();

// Optional: validate MAC address format

if (!macAddress.matches("(?i)([0-9A-F]{2}[:-]){5}[0-9A-F]{2}")) {

System.out.println("Invalid MAC address format.");

client.close();

return;

}

// Send MAC address to server

out.println(macAddress);

// Read IP address (response) from server

String response = in.readLine();

// Display result

if (response == null || response.isEmpty()) {

System.out.println("No response from server.");

} else if (response.toLowerCase().contains("not found")) {

System.out.println("IP address not found for given MAC.");

} else {

System.out.println("IP Address: " + response);

}

// Close resources

out.close();

in.close();

userInput.close();

client.close();

} catch (IOException e) {

System.err.println("Client error: " + e.getMessage());

}

}

}

**Output**

Server

$ javac RarpServer.java

$ java RarpServer Server started

Client

$ javac RarpClient.java

$ java RarpClient

Enter the physical address : B8:AC:6F:1B:AB:06

Logical Address 172.16.12.251

Result

Thus using RARP protocol, IP address of the server is obtained