ARP AND RARP

#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <sys/shm.h>

#include <string.h>

#include <unistd.h>

#define SHM\_KEY 3000

#define MAX\_ENTRIES 3

#define NAME\_LENGTH 50

#define IP\_LENGTH 16

#define SHM\_SIZE (MAX\_ENTRIES \* (NAME\_LENGTH + IP\_LENGTH + 2)) // Name + IP + newline + space

int main() {

int shmid, a, i;

char \*ptr, \*shmptr;

// Create a shared memory segment

shmid = shmget(SHM\_KEY, SHM\_SIZE, IPC\_CREAT | 0666);

if (shmid < 0) {

perror("shmget failed");

exit(1);

}

// Attach the shared memory segment

shmptr = shmat(shmid, NULL, 0);

if (shmptr == (char \*)(-1)) {

perror("shmat failed");

exit(1);

}

ptr = shmptr;

// Collect entries

for (i = 0; i < MAX\_ENTRIES; i++) {

puts("Enter the name:");

scanf("%s", ptr);

a = strlen(ptr);

ptr[a] = ' '; // Add space after the name

puts("Enter IP:");

ptr = ptr + a + 1; // Move pointer to the next position

scanf("%s", ptr);

ptr[strlen(ptr)] = '\n'; // Add newline after the IP

ptr += strlen(ptr) + 1; // Move pointer for the next entry

}

// Null terminate the string

\*ptr = '\0';

// Print ARP table

printf("\nARP table at server side is:\n%s", shmptr);

// Detach the shared memory

shmdt(shmptr);

return 0;

}

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <sys/shm.h>

#include <unistd.h>

#define SHM\_KEY 3000

#define MAX\_ENTRIES 3

#define NAME\_LENGTH 50

#define IP\_LENGTH 16

#define MAC\_LENGTH 26

#define SHM\_SIZE (MAX\_ENTRIES \* (NAME\_LENGTH + IP\_LENGTH + 2)) // Name + IP + newline + space

int main() {

int shmid, a;

char \*ptr, \*shmptr;

char ptr2[NAME\_LENGTH], ip[IP\_LENGTH], mac[MAC\_LENGTH];

// Access the shared memory segment

shmid = shmget(SHM\_KEY, SHM\_SIZE, 0666);

if (shmid < 0) {

perror("shmget failed");

exit(1);

}

// Attach the shared memory segment

shmptr = shmat(shmid, NULL, 0);

if (shmptr == (char \*)(-1)) {

perror("shmat failed");

exit(1);

}

// Print the ARP table

puts("The ARP table is:");

printf("%s\n", shmptr);

while (1) {

printf("\n1. ARP\n2. RARP\n3. EXIT\n");

printf("Choose an option: ");

scanf("%d", &a);

switch (a) {

case 1: // ARP

puts("Enter IP address:");

scanf("%s", ip);

ptr = strstr(shmptr, ip);

if (ptr != NULL) {

ptr -= 8; // Adjust pointer to MAC address

sscanf(ptr, "%s%\*s", ptr2); // Read MAC address

printf("MAC address is: %s\n", ptr2);

} else {

printf("IP address not found.\n");

}

break;

case 2: // RARP

puts("Enter MAC address:");

scanf("%s", mac);

ptr = strstr(shmptr, mac);

if (ptr != NULL) {

sscanf(ptr, "%\*s%s", ptr2); // Read IP address

printf("IP address is: %s\n", ptr2);

} else {

printf("MAC address not found.\n");

}

break;

case 3: // EXIT

shmdt(shmptr); // Detach shared memory before exit

exit(0);

default:

printf("Invalid option. Please try again.\n");

}

}

// Detach the shared memory (in case of exiting from loop)

shmdt(shmptr);

return 0;

}