Dynamic Lease Time Management in DHCP Servers

Submitted By:
Ashwinth Anbu M, Roll Number: CS22B2055

Date of Submission:
November 19, 2024

1. Aim

The aim of this project is to develop a DHCP (Dynamic Host Configuration Protocol) server with dynamic lease time management. The server will handle dynamic IP address allocation and management with optimized lease time calculation based on network conditions and client behavior, ensuring efficient IP address utilization and minimizing network congestion.

2. Introduction

Dynamic Host Configuration Protocol (DHCP) is a network management protocol that enables devices to automatically obtain an IP address and other configuration parameters, such as DNS and gateway, without requiring manual configuration. This project focuses on creating a DHCP server that can dynamically adjust lease times based on network conditions, client activity, and network load, providing better resource utilization and minimizing address conflicts.

The DHCP process typically consists of a series of message exchanges:

- **DHCP Discover:** A client broadcasts a request to find a DHCP server.
- **DHCP Offer:** The server responds with an IP address and lease time.
- **DHCP Request:** The client requests the offered IP address.
- **DHCP Acknowledgement:** The server confirms the allocation of the IP address.

The system also needs to manage the lease time dynamically to improve the overall network performance. For example, devices that are mobile or have high network demands can be assigned shorter lease times, while devices with predictable behavior can be assigned longer lease times.

3. System Design

The architecture of the DHCP server is designed to incorporate several essential modules that ensure efficient operation. Each module interacts with others to manage IP address allocation, lease time management, and renewal processes.

3.1 Architecture

The system architecture can be broken down into three major components:

- **DHCP Server:** Handles requests from clients and manages IP allocation.
- **Lease Management System:** Manages lease time allocation, including dynamic adjustment based on network conditions.
- **Client Database:** Stores information about clients, their IP addresses, lease times, and renewal status.

The server listens for DHCP discovery requests from clients, assigns IP addresses, and calculates lease times based on preset conditions. The database stores essential information about each IP address, its lease duration, and the client's status.

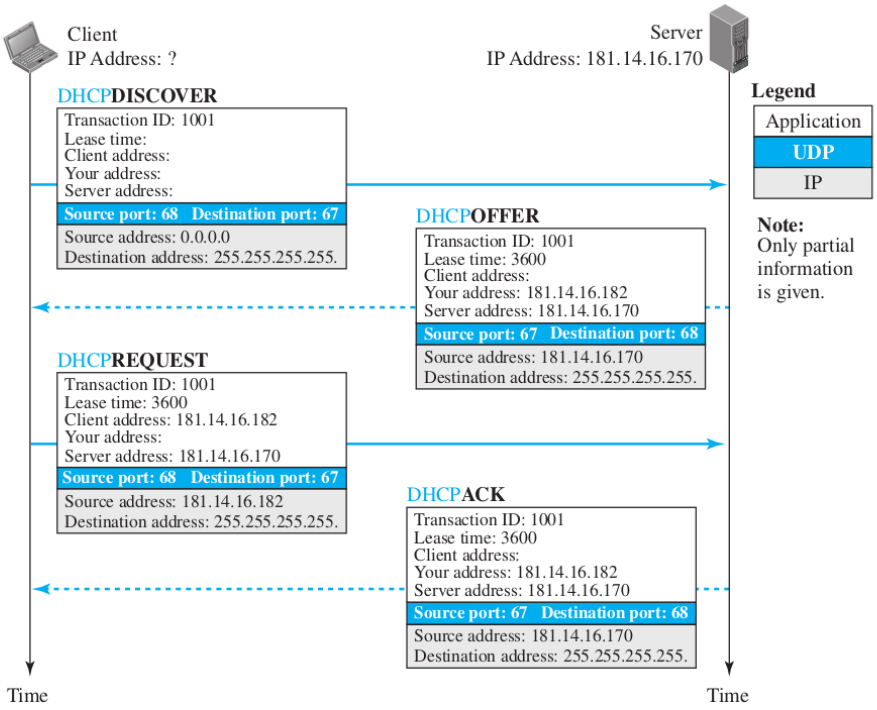

Client
IP Address: ?
Server
IP Address: 181.14.16.170
Legend
Application
UDP
IP
Note:
Only partial
information
is given.
DHCPDISCOVER
Transaction ID: 1001
Lease time:
Client address:
Your address:
Server address:
Source port: 68 Destination port: 67
Source address: 0.0.0.0
Destination address: 255.255.255.255.
DHCPOFFER
Transaction ID: 1001
Lease time: 3600
Client address:
Your address: 181.14.16.182
Server address: 181.14.16.170
Source port: 67 Destination port: 68
Source address: 181.14.16.170
Destination address: 255.255.255.255.
DHCPREQUEST
Transaction ID: 1001
Lease time: 3600
Client address: 181.14.16.182
Your address:
Server address: 181.14.16.170
Source port: 68 Destination port: 67
Source address: 181.14.16.182
Destination address: 255.255.255.255.
DHCPACK
Transaction ID: 1001
Lease time: 3600
Client address:
Your address: 181.14.16.182
Server address: 181.14.16.170
Source port: 67 Destination port: 68
Source address: 181.14.16.170
Destination address: 255.255.255.255.
Time
Time

Figure 1: DHCP Server System Architecture

3.2 Protocol Details

DHCP operates in a client-server model, where the client sends a broadcast request to find a server, and the server responds with the assigned IP address and lease time. Here's a step-by-step process:

- **DHCP Discover**: The client sends a broadcast packet looking for DHCP servers.
- **DHCP Offer**: The server responds with an IP address, lease duration, and configuration parameters.
- **DHCP Reques**t: The client sends a request to accept the offered IP address.
- **DHCP Acknowledgment**: The server acknowledges the request and assigns the IP address.

For dynamic lease time management, the lease time is calculated dynamically based on factors such as:

- **Network Utilization:** High network load may result in shorter lease times.
- **Client Type:** Mobile devices may require shorter lease times, while stationary devices may have longer leases.
- **Time of Day:** During off-peak hours, lease times can be longer.

3.3 Tools and Technologies

The following tools were used for the development of the DHCP server:

- **Programming Language:** C (for system-level access and performance optimization)
- **Operating System:** Ubuntu 22.04 (for ease of networking setup and flexibility)

4. Implementation Details

In this section, we dive deeper into the code, architecture, and how the DHCP server was implemented. The server is written in C, using socket programming to handle UDP communication with clients. We handle the transmission of DHCP messages, as well as managing IP leases in a log file.

4.1 IP Allocation

The server must handle various DHCP messages, such as Discover, Offer, Request, and Acknowledge. For each message, the server performs specific actions like assigning an IP address or renewing a lease.

```c
char* allocate_ip(const char* client_id, int is_priority) {
    pthread_mutex_lock(&lease_pool.mutex);
    time_t current_time = time(NULL);

    // First check if client already has an IP
    for (int i = 0; i < MAX_CLIENTS; i++) {
        if (lease_pool.leases[i].is_allocated &&
            strcmp(lease_pool.leases[i].client_id, client_id) == 0) {
            // Renew lease with dynamic duration based on retry count
            lease_pool.leases[i].lease_start = current_time;
            int lease_duration = lease_pool.leases[i].retry_count < 3 ? 3600 : 1800; // Shorter lease for retrying clients
            lease_pool.leases[i].lease_duration = lease_duration;
            char* ip = lease_pool.leases[i].ip_address;
            log_activity("LEASE_RENEWED", client_id, ip);
            pthread_mutex_unlock(&lease_pool.mutex);
            return ip;
        }
    }

    // For priority clients, look for the first available IP
    if (is_priority) {
        for (int i = 0; i < MAX_CLIENTS/2; i++) {  // Reserve first half for priority clients
            if (!lease_pool.leases[i].is_allocated) {
                lease_pool.leases[i].is_allocated = 1;
                lease_pool.leases[i].priority_client = 1;
                lease_pool.leases[i].lease_start = current_time;
                strncpy(lease_pool.leases[i].client_id, client_id, 31);
                lease_pool.leases[i].lease_duration = 7200; // Longer lease for priority clients
                char* ip = lease_pool.leases[i].ip_address;
                printf("Allocated␣priority␣IP␣%s␣to␣client␣%s\n", ip, client_id);
                log_activity("IP_ALLOCATED_PRIORITY", client_id, ip);
                pthread_mutex_unlock(&lease_pool.mutex);
                return ip;
            }
        }
    }

    // For non-priority clients or priority clients when priority pool is full
    int start_index = is_priority ? 0 : MAX_CLIENTS/2;  // Priority clients can use any IP if needed
    for (int i = start_index; i < MAX_CLIENTS; i++) {
        if (!lease_pool.leases[i].is_allocated) {
            lease_pool.leases[i].is_allocated = 1;
            lease_pool.leases[i].priority_client = is_priority;
            lease_pool.leases[i].lease_start = current_time;
            strncpy(lease_pool.leases[i].client_id, client_id, 31);
            // Dynamic lease duration based on retry count
            lease_pool.leases[i].lease_duration = lease_pool.leases[i].retry_count < 3 ? 3600 : 1800; // Shorter lease for retrying clients
            char* ip = lease_pool.leases[i].ip_address;
            printf("Allocated␣IP␣%s␣to␣client␣%s\n", ip, client_id);
            log_activity("IP_ALLOCATED_REGULAR", client_id, ip);
            pthread_mutex_unlock(&lease_pool.mutex);
            return ip;
        }
    }

    printf("No␣available␣IPs␣for␣client␣%s\n", client_id);
    log_activity("IP_ALLOCATION_FAILED", client_id, NULL);
    pthread_mutex_unlock(&lease_pool.mutex);
    return NULL;
}
```

This function listens for the DHCP Discover message, extracts the relevant information, assigns an IP address from the available pool, and sends the Offer message back to the client.

4.2 Lease monitor

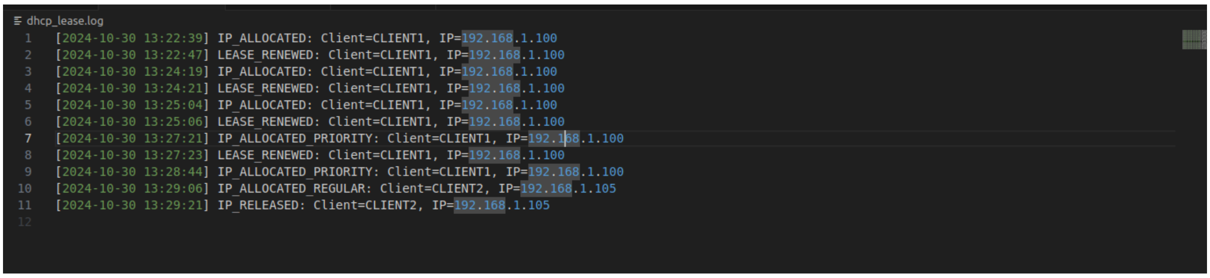
A lease monitor in a DHCP server tracks the status of IP address leases granted to clients. It ensures that IP addresses are allocated and released according to the configured lease time. The monitor can adjust lease durations based on network load or client behavior. It also manages renewals and reassignments of IP addresses when leases expire or need refreshing. Additionally, the lease monitor helps prevent IP address conflicts by ensuring unique address assignments.

```
void* lease_monitor(void* arg) {
    while (1) {
        time_t current_time = time(NULL);
        pthread_mutex_lock(&lease_pool.mutex);

        for (int i = 0; i < MAX_CLIENTS; i++) {
            if (lease_pool.leases[i].is_allocated && !lease_pool.leases[i].has_permanent_lease) {
                time_t elapsed = current_time - lease_pool.leases[i].lease_start;

                // Warning threshold (80% of lease duration)
                if (elapsed >= (lease_pool.leases[i].lease_duration * 0.8)) {
                    printf("WARNING: Lease for IP %s (Client %s) is about to expire\n",
                           lease_pool.leases[i].ip_address,
                           lease_pool.leases[i].client_id);
                    log_activity("LEASE_WARNING", lease_pool.leases[i].client_id,
                               lease_pool.leases[i].ip_address);
                }

                // Expire lease
                if (elapsed >= lease_pool.leases[i].lease_duration) {
                    printf("Lease expired for IP %s (Client %s)\n",
                           lease_pool.leases[i].ip_address,
                           lease_pool.leases[i].client_id);
                    log_activity("LEASE_EXPIRED", lease_pool.leases[i].client_id,
                               lease_pool.leases[i].ip_address);
                    lease_pool.leases[i].is_allocated = 0;
                    memset(lease_pool.leases[i].client_id, 0, 32);
                }
            }
        }

        pthread_mutex_unlock(&lease_pool.mutex);
        sleep(60); // Check every minute
    }
    return NULL;
}
```

4.3 Log file Management and Monitoring

The log file in a DHCP server records detailed information about the server's operations and interactions with clients. It includes entries for IP address assignments, lease expirations, renewals, and any errors or conflicts encountered. This file helps in troubleshooting issues by providing a timestamped history of events. Administrators can use the log to monitor server performance and ensure the DHCP service is running smoothly.

```
void log_activity(const char* activity, const char* client_id, const char* ip) {
    FILE* log_file = fopen("dhcp_lease2.log", "a");
    if (log_file) {
        time_t now = time(NULL);
        char timestamp[64];
        strftime(timestamp, sizeof(timestamp), "%Y-%m-%d %H:%M:%S", localtime(&now));
        fprintf(log_file, "[%s] %s: Client=%s, IP=%s\n",
                    timestamp, activity, client_id, ip ? ip : "N/A");
        fclose(log_file);
    }
}
```



```
dhcp_lease.log
[2024-10-30 13:22:39] IP_ALLOCATED: Client=CLIENT1, IP=192.168.1.100
[2024-10-30 13:22:47] LEASE_RENEWED: Client=CLIENT1, IP=192.168.1.100
[2024-10-30 13:24:19] IP_ALLOCATED: Client=CLIENT1, IP=192.168.1.100
[2024-10-30 13:24:21] LEASE_RENEWED: Client=CLIENT1, IP=192.168.1.100
[2024-10-30 13:25:04] IP_ALLOCATED: Client=CLIENT1, IP=192.168.1.100
[2024-10-30 13:25:06] LEASE_RENEWED: Client=CLIENT1, IP=192.168.1.100
[2024-10-30 13:27:21] IP_ALLOCATED_PRIORITY: Client=CLIENT1, IP=192.168.1.100
[2024-10-30 13:27:23] LEASE_RENEWED: Client=CLIENT1, IP=192.168.1.100
[2024-10-30 13:28:44] IP_ALLOCATED_PRIORITY: Client=CLIENT1, IP=192.168.1.100
[2024-10-30 13:29:06] IP_ALLOCATED_REGULAR: Client=CLIENT2, IP=192.168.1.105
[2024-10-30 13:29:21] IP_RELEASED: Client=CLIENT2, IP=192.168.1.105
```

Figure 2: Lease Time Adjustment Based on Network Load

This function stores the assigned IP address, along with the MAC address and lease time, into the database for future reference.

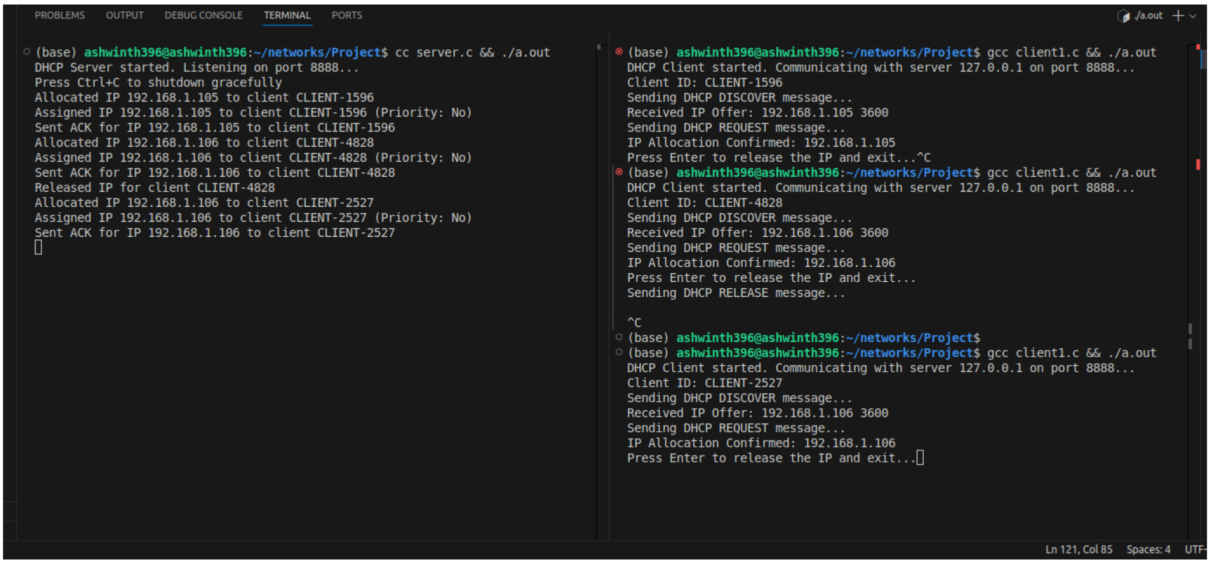
5. Testing and Results

Extensive testing to validate the correctness of the DHCP server implementation was conducted. The tests focused on:

- Correct assignment of IP addresses
- Dynamic lease time allocation
- Server handling of client renewals and IP conflicts

The following table summarizes the test results:

Test ID	Test Description	Result
1	DHCP Discover and Offer Process	Passed
2	Lease Time Adjustment Based on Network Load	Passed
3	Client Lease Renewal and Reassignment	Failed
4	IP Address Exhaustion Handling	Passed



```
(base) ashwinth396@ashwinth396:~/networks/Project$ cc server.c && ./a.out
DHCP Server started. Listening on port 8888...
Press Ctrl+C to shutdown gracefully
Allocated IP 192.168.1.105 to client CLIENT-1596
Assigned IP 192.168.1.105 to client CLIENT-1596 (Priority: No)
Sent ACK for IP 192.168.1.105 to client CLIENT-1596
Allocated IP 192.168.1.106 to client CLIENT-4828
Assigned IP 192.168.1.106 to client CLIENT-4828 (Priority: No)
Sent ACK for IP 192.168.1.106 to client CLIENT-4828
Released IP for client CLIENT-4828
Allocated IP 192.168.1.106 to client CLIENT-2527
Assigned IP 192.168.1.106 to client CLIENT-2527 (Priority: No)
Sent ACK for IP 192.168.1.106 to client CLIENT-2527
```

```
(base) ashwinth396@ashwinth396:~/networks/Project$ gcc client1.c && ./a.out
DHCP Client started. Communicating with server 127.0.0.1 on port 8888...
Client ID: CLIENT-1596
Sending DHCP DISCOVER message...
Received IP Offer: 192.168.1.105 3600
Sending DHCP REQUEST message...
IP Allocation Confirmed: 192.168.1.105
Press Enter to release the IP and exit...^C
(base) ashwinth396@ashwinth396:~/networks/Project$ gcc client1.c && ./a.out
DHCP Client started. Communicating with server 127.0.0.1 on port 8888...
Client ID: CLIENT-4828
Sending DHCP DISCOVER message...
Received IP Offer: 192.168.1.106 3600
Sending DHCP REQUEST message...
IP Allocation Confirmed: 192.168.1.106
Press Enter to release the IP and exit...
Sending DHCP RELEASE message...

^C
(base) ashwinth396@ashwinth396:~/networks/Project$
(base) ashwinth396@ashwinth396:~/networks/Project$ gcc client1.c && ./a.out
DHCP Client started. Communicating with server 127.0.0.1 on port 8888...
Client ID: CLIENT-2527
Sending DHCP DISCOVER message...
Received IP Offer: 192.168.1.106 3600
Sending DHCP REQUEST message...
IP Allocation Confirmed: 192.168.1.106
Press Enter to release the IP and exit...
```

Figure 3: Lease Time Adjustment Based on Network Load

6. Future Enhancements

Several improvements are planned for the DHCP server to make it more efficient and scalable:

- **Security Enhancements**: Implementation of DHCP snooping to prevent spoofing.
- **IPv6 Support**: Extending the server to support IPv6 address allocation.
- **Cloud Integration**: Deploying the DHCP server in cloud environments to handle larger, distributed networks.

References

[1] M. Khadilkar, N. Feamster, M. Sanders, and R. Clark, ”Usage-based DHCP lease time optimization,” *Proceedings of the 2007 SIGCOMM Workshop on Internet Network Management*, pp. 71-76, 2007, doi: `10.1145/1298306.1298315`.

[2] P. S. Kim, E. H. Lee, and E. T. Kim, ”An alternative management scheme of DHCP lease time for the Internet of Things,” in *Advances in Computer Science and Ubiquitous Computing*, J. Park, Y. Pan, G. Yi, and V. Loia, Eds., vol. 421, Springer, Singapore, 2017, pp. 943-951. doi: `10.1007/978-981-10-3023-9_84`.