

# HS DHCP Server C Source Library v1.1

## User Manual

29 March 2012  
Rev 1.1

### Table of Contents

1 Introduction.....	2
1.1 About DHCP Protocol.....	2
1.2 About HS DHCP Library.....	2
2 HS DHCP API.....	3
2.1.1 HsDhcpInit.....	3
2.1.2 HsDhcpCleanUp.....	3
2.1.3 HsDhcpGetLocalInterfaceList.....	4
2.1.4 HsDhcpServerStart.....	5

# 1 Introduction

## 1.1 About DHCP Protocol

The Dynamic Host Configuration Protocol (DHCP) is an autoconfiguration protocol used on IP networks. Computers that are connected to IP networks must be configured before they can communicate with other computers on the network. DHCP allows a computer to be configured automatically, eliminating the need for intervention by a network administrator. It also provides a central database for keeping track of computers that have been connected to the network. This prevents two computers from accidentally being configured with the same IP address.

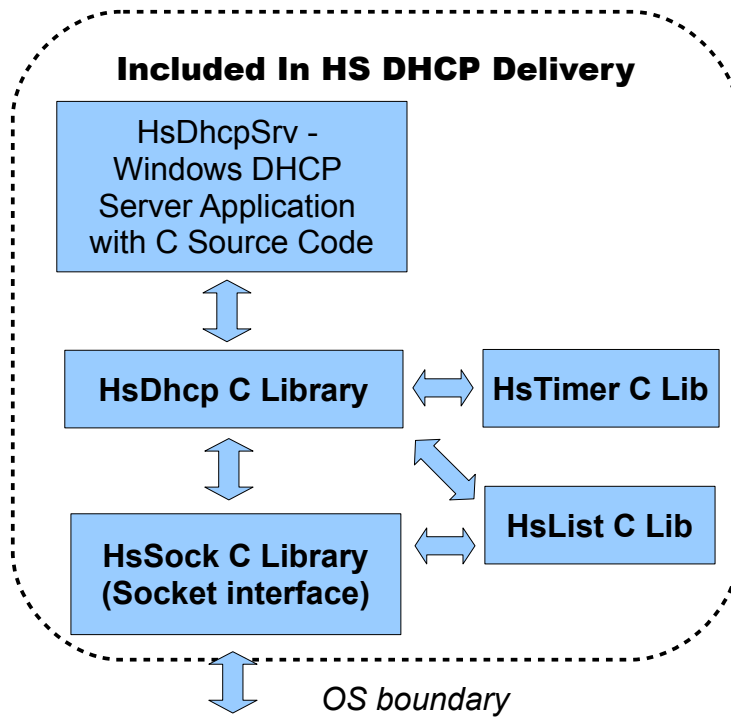
## 1.2 About HS DHCP Library

HS DHCP is a DHCP Server library implemented in C language. HS DHCP implements the server part of DHCP as per RFC 2131.

The HS DHCP product delivery includes a simple DHCP Server application (HsDhcpSrv) for Windows using HS DHCP library.

HS DHCP is delivered to the user with the following components:

- HS DHCP C source library implements core DHCP server functionality
- HsDhcpSrv – windows server application with C source code which uses HsDhcp
- HsTimer C library – provides HsDhcp with the necessary timer functionality
- HsList C Library, list management library
- HsSock C Library – socket interface library



### HS DHCP Features:

Compliance to RFC standards	RFC2131
Server support	yes
Client support	No
Number of supported DHCP clients	253
Bind to different local IP interfaces	Yes
Supports DHCP messages	DISCOVER, REQUEST, DECLINE, RELEASE, OFFER, ACK, NACK
Supports DHCP options	Lease time, rebind time, renew time, subnet mask, server id, client id, parameter list, domain name, router ip, DNS server name, netbios server ip
Tracing / event log	yes

## 2 HS DHCP API

To call HS DHCP API functions include the following header file:

HsDhcp\_if.h

### 2.1.1 HsDhcpInit

Initializes HS DHCP library. Call this function once at program initialization, before any other functions are called.

#### Syntax:

```
int HsDhcpInit(char *errstr);
```

#### Parameters:

\*errstr – pointer to error string buffer that will receive error description string in case the function fails.

#### Return values:

If the function succeeds the return value is 1. If the function fails, the return value is 0 and the descriptive null terminated ASCII error description string is copied into the buffer pointed to by \*errstr.

#### Remarks:

This function must be called before any other functions of the library are called. Call this function once at the startup of your application. On application exit, call HsDhcpCleanUp to stop any running services and de-allocate the resources used by HS DHCP library.

### 2.1.2 HsDhcpCleanUp

De-Initializes HS DHCP library. Call this function once at your program exit.

#### Syntax:

```
void HsDhcpCleanUp(void);
```

#### Parameters:

none.

#### Return values:

none.

#### Remarks:

This function must be called before you exit your application to cleanup and release all resources used by HS DHCP library.

## 2.1.3 HsDhcpGetLocalInterfaceList

Obtains the list of local IP interface addresses.

### Syntax:

```
hsdhcp_if_list_item_t *HsDhcpGetLocalInterfaceList(void);
```

### Parameters:

none.

### Return values:

The pointer to the linked list of addresses is returned in case of success. NULL is returned in case of error.

each element item of type `hsdhcp_if_list_item_t` has the following structure

- `index` - zero based interface number
- `ipaddr_str` - zero terminated ip address string (e.g. "192.168.1.1")
- `*next` - pointer to the next element, or NULL if it is the end of the list

```
// local interface list element
typedef struct
{
    int                index;
    unsigned char     ipaddr_str[16];
    void              *next;
} hsdhcp_if_list_item_t;
```

### Remarks:

Use this function to obtain a list of local IP interface addresses available. Then you can start HS DHCP (see function `HsDhcpServerStart`) on a specific local IP interface identified by interger, zero based interface index.

## 2.1.4 HsDhcpServerStart

Obtains the list of local IP interface addresses.

### Syntax:

```
int HsDhcpServerStart(hsdhcp_server_params_t *pParams, char *errstr);
```

### Parameters:

\*errstr – pointer to error string buffer that will receive error description string in case the function fails.

\*pParams – pointer to server initialization parameters structure of type hsdhcp\_server\_params\_t, defined below:

Data member	description
int local_if_index;	Local IP interface index, use 0 to use the default local interface, otherwise use a required index as obtained by calling HsDhcpGetLocalInterfaceList
unsigned char ipaddr_start[16];	First ip address in the DHCP range that server will allocate to clients (in dotted format, zero terminated eg 192.168.1.2)
unsigned char ipaddr_end[16];	Last ip address in the DHCP range that server will allocate to clients (in dotted format, zero terminated eg 192.168.1.254)
unsigned char subnet_mask[16];	Subnet mask of the served DHCP clients – use 255.255.255.0
unsigned char router_ip[16];	Router IP address to of the router serving DHCP clients to be sent to DHCP clients if they request it
unsigned char dns_ip[16];	DNS server IP address to be sent to DHCP clients if they request it
unsigned char servername[64];	This DHCP server name, this parameter is sent to DHCP clients if they request it
unsigned char domain_name[256];	Network Domain name, this parameter is sent to DHCP clients if they request it
int log_level;	log level 0 = disabled, 9=most detailed, debug level.
log_fn_t *log_fn;	event log function callback. HS DHCP calls this function to log events if log_level is not 0.  // event log function prototype typedef void log_fn_t(char *str);
TimerStart_cb_t*timerStart_fn;	Timer start callback function  // Start timer // timeout_ms - timeout in milliseconds // pData - user data to pass to timeout callback function // cb - timeout callback function, called on timer expiry // returns timer handle or NULL if timer could not be started typedef void *TimerStart_cb_t(unsigned long timeout_ms, void *pData, hs_timer_cb_t *cb);  // timer callback prototype, called on timer expiry typedef void hs_timer_cb_t(long lData);
TimerStop_cb_t*timerStop_fn;	Timer stop callback function  // pTimerHan - timer handle of the timer to stop typedef void TimerStop_cb_t(void *pTimerHan);
AddBinding_cb_t *addBinding_fn;	Add binding callback function - callback function to add a new client IP address binding into persistent storage

	<pre>typedef void AddBinding_cb_t(hsdhcp_client_identifier_t *pCLID, unsigned long ullpaddr);  // client identifier typedef struct {     unsigned char len; // client identifier data length     unsigned char buf[256]; // client identifier data } hsdhcp_client_identifier_t;</pre>
GetBindingIPAddr_cb_t *getBinding_fn	<p>Get binding callback function - callback function to lookup client IP address binding from persistent storage by client identifier</p> <pre>typedef unsigned long GetBindingIPAddr_cb_t(hsdhcp_client_identifier_t *pCLID);  // client identifier typedef struct {     unsigned char len; // client identifier data length     unsigned char buf[256]; // client identifier data } hsdhcp_client_identifier_t;</pre>
hsdhcp_ev_callback_t *ev_callback_fn	<p>User event callback function, used to report events to user application.</p> <pre>typedef void hsdhcp_ev_callback_t(int local_if_index, int ev);</pre>

### Return values:

If the function succeeds the return value is 1. If the function fails, the return value is 0 and the descriptive null terminated ASCII error description string is copied into the buffer pointed to by \*errstr.

### Remarks:

Use this function to start DHCP library in server mode. If the function succeeds, it starts serving DHCP clients.

local\_if\_index parameter is used to specify which local interface the server should run on. The list of local ip addresses available can be obtained by calling HsDhcpGetLocalInterfaceList. The applicatoin can use local\_if\_index=0 to use first available local ip interface.

The user application must implement a number of callback functions in its code to provide HS DHCP with required services: event loggin, timer services, DHCP client database / persistent storage and lookup.

Function callback addBinding\_fn should add an element consisting of client identifier (currently MAC address) and the allocated IP address to persistent storage database. The HsDhcpSrv example uses a simple binary disk file storage for this purpose.

Function callback getBinding\_fn should implement the lookup of the IP address by client identifier in the persistent storage.

According to RFC2131 this mechanism is used to allocate as must as possible consistently the same IP addresses to the same DHCP clients (having the same client identifiers - MAC addresses)

Function callback ev\_callback\_fn currently is used by HS DHCP to report a single event HSDHCP\_USR\_EV\_CLOSED in case a UDP socket on which the server is running has closed unexpectedly. The applicatoin can process this event and take actions, such as restart the server.