mDNS is a protocol that provides DNS like facility to advertise and discover services in a local area network. The mDNS based service discovery is defined by two RFCs, 6762 and 6763.

# Features and Limitations

Following is the list of features provided by this component:

1. Support for IPV4
2. Register hostname
3. Register services
4. Responding to Multi-Question queries
5. Known-Answer Suppression
6. Respond to ADDR, SRV and ANY type of queries
7. One-Shot Multicast DNS queries

Following is the list of some limitations,

1. IPV6 is not supported
2. Probing and conflict detection is not supported
3. Continues multicast querying

# Header file/s

Components/mdns/inc/mdns.h.

# Data Structure Definitions

## mdns\_srvc\_info\_t

This data structure is used to pass information about the discovered service. Information is passed through the callback function provided while calling the mdns\_service\_discover API.

|  |  |
| --- | --- |
| ***instance*** | Service Instance name. max limit 63 bytes. Example: “innoprov” |
| ***srvc\_type*** | Service protocol. Example: “\_tcp” |
| ***protocol*** | Service protocol. Example: “\_tcp” |
| ***domain*** | Domain. In almost all the cases its “local”. This is the default value |
| ***txtInfo*** | Txt record information data |
| ***port*** | Port on which service is available |
| ***ipaddr*** | IP address of the interface on which service is available |

Table : mdns\_srvc\_info\_t - parameters

## mdns\_hnreg\_param\_t

This data structure is used to pass parameters while registering the Host Name. Following are some of the important parameters:

|  |  |
| --- | --- |
| ***hostname*** | Host name of the device. Keep this unique in the N/w. Best done by using mac address as part of the host name |
| ***domain*** | Always pass MDNS\_LOCAL\_DOMAIN |
| ***ipaddr*** | IP address of the host interface in little endian format |
| ***ttl*** | Time to live. Default value is 120 sec |

Table : mdns\_hnreg\_param\_t - parameters

## mdns\_srvreg\_param\_t

This data structure is used to pass the parameters while registering a service. Following are some of the important parameters:

|  |  |
| --- | --- |
| ***srvc\_name*** | Service name. Example: “prov” |
| ***srvc\_sub\_type*** | Name of the subtype excluding "\_sub". "\_sub" is added implicitly |
| ***srvc\_type*** | Type of service. Example: "\_http" |
| ***srvc\_proto*** | Protocol type. "\_tcp" or "\_udp" |
| ***domain*** | Always pass MDNS\_LOCAL\_DOMAIN |
| ***txt\_data*** | Pointer to key/value pairs conveying additional information about the named service |
| ***port*** | Port number for the given service protocol type |
| ***ttl*** | Time to live. Default 120 sec |

Table : mdns\_srvreg\_param\_t – parameters

## mdns\_srvcdisc\_param\_t

This data structure is used for passing the parameters while discovering for service of specific type.

|  |  |
| --- | --- |
| ***srvc\_sub\_type*** | Name of the subtype excluding "\_sub". "\_sub" is added implicitly |
| ***srvc\_type*** | Type of service. Example: "\_http" |
| ***srvc\_proto*** | Protocol type. "\_tcp" or "\_udp" |
| ***domain*** | Always pass MDNS\_LOCAL\_DOMAIN |
| ***cb*** | mDNS service discovery callback function |
| ***cbctx*** | Callback context to be passed to cb |

Table : mdns\_srvcdisc\_param\_t - parameters

# API Reference

## mdns\_init

### Overview

This function is used for initializing the mDNS component. This returns pointer to mDNS handle. This needs to be passed to all other APIs. This shall be a onetime initialization.

Apart from other initializations, this call creates a thread and a message queue for handling mDNS activities like packet parsing and timeouts.

### Definition

|  |
| --- |
| mdns\_ctx\_t \*  mdns\_init(mdns\_init\_params\_t \*param); |

### Parameters

|  |  |
| --- | --- |
| **Parameters** | **Description** |
| *param* | Pointer to data structure of type mdns\_init\_params\_t |

### Return

Success: Pointer to mDNS handle

Error: NULL

## mdns\_hostname\_reg

### Overview

This function is used for registering the host name of the interface(node).

### Definition

|  |
| --- |
| int  mdns\_hostname\_reg(mdns\_ctx\_t \*mc, mdns\_hnreg\_param\_t \*param); |

### Parameters

|  |  |
| --- | --- |
| **Parameters** | **Description** |
| *mc* | Pointer returned by mdns\_init() |
| *param* | Pointer to structure of type mdns\_hnreg\_param\_t |

### Return

Success: 0

Error: -1

## mdns\_service\_register

### Overview

This function is used for registering the service. Once registered, response for any matching query will be given internally.

### Definition

|  |
| --- |
| int  mdns\_service\_register(mdns\_ctx\_t \*mc, mdns\_srvreg\_param\_t \*param) |

### Parameters

|  |  |
| --- | --- |
| **Parameters** | **Description** |
| *mc* | Pointer returned by mdns\_init() |
| *param* | Pointer to structure of type mdns\_srvreg\_param\_t |

### Return

Success: 0

Error: -1

## mdns\_service\_discover

### Overview

This function is used for discovering services of given type. When service is discovered, callback function specified in service discovery parameter structure will be called to notify the application.

Callback function is called separately for each service it discovers. If more than one services are detected, callback function will be called as many times as the total number of service detected.

### Definition

|  |
| --- |
| int  mdns\_service\_discover(mdns\_ctx\_t \*mc, mdns\_srvcdisc\_param\_t \*param) |

### Parameters

|  |  |
| --- | --- |
| **Parameters** | **Description** |
| *mc* | Pointer returned by mdns\_init() |
| *param* | Pointer to structure of type mdns\_srvcdisc\_param\_t |

### Return

Success: 0

Error: -1

# Application Example

For the example code, refer: *examples/mdns application*.