Technical Q&A NW61

Accessing DHCP Options

Technical Q&A

NW 61 - Accessing DHCP Options (19-April-99)

Q: I'm writing a cross-platform connectivity product. One of the features of the product is that it can automatically configure itself based on extra options returned in the DHCP configuration. Is there any way I can access these extra options from the DHCP packet that OT used to configure the TCP/IP stack?

A: Prior to Open Transport 2.0.1, there was no way to access the extra DHCP options that were in the DHCP packet OT used to configure the TCP/IP stack. Open Transport 2.0.1 introduces a new API routine, `OTInetGetDHCPConfigInfo`, which allows access to all options in the DHCPACK packet used to configure the TCP/IP stack. This Q&A describes the new routine.

Note:
To use this routine you will need the interfaces and libraries from the Open Transport 2.0.1 SDK (or later).

InetDHCPOption

```
struct InetDHCPOption
{
    UInt8       fOptionTag;
    UInt8       fOptionLen;
    UInt8       fOptionValue;
};
typedef struct InetDHCPOption InetDHCPOption;
```

This structure is used to return information about a DHCP option. The fields have the following meaning:

`fOptionTag`
This is the DHCP **tag octet** as defined in RFC 2132 DHCP Options and BOOTP Vendor Extensions.

`fOptionLen`
This is the total length of the option, not including the size of the tag octet and this length byte.

`fOptionValue`
This field is a place holder for an unbounded array of bytes that make up the option.

OTInetGetDHCPConfigInfo

```
extern pascal OSStatus OTInetGetDHCPConfigInfo(InetDHCPOption* buf,
                    UInt32 bufSize, SInt32 index, SInt32 opt);
```

buf	A pointer to a buffer where OT places the DHCP option.
bufSize	The size of the above buffer.
index	The interface from whose DHCP configuration you wish to extract options. This is the same index used to specify an interface when calling `OTInetGetInterfaceInfo`. Specify `kDefaultInetInterface` if you wish to get options from the default interface.
opt	The tag octet of the option to be extracted from the DHCP configuration, or `kAllDHCPOptions` to receive all options. If you ask for all options, they are packed in the same way as they are in the DHCP configuration.
result	An error result. Likely errors include `kENOENTErr` (TCP/IP is not loaded, or the interface specified by index is not found or not loaded), `kEINVALErr` (`index` is `kDefaultInetInterface` but the default interface isn't configured, or the specified interface wasn't configured using DHCP or BOOTP), `kENORSRCErr` (DHCP information is not available because there was not enough memory to save it when the TCP/IP stack loaded), `kOTBufferOverflowErr` (explained below), or `kOTBadOptionErr` (the option specified by `opt` was not in the DHCP configuration).

This routine returns, in the buffer described by `buf` and `bufSize`, the value of the DHCP option for the interface specified by `index` whose tag octet is `opt`. You can specify `kDefaultInetInterface` for `index` to get the options for the default interface. You can specify `kAllDHCPOptions` for `opt` to get all DHCP options for an interface.

The standard values for `opt` are specified in RFC 1700 Assigned Numbers. The format of the returned option values for these options are defined in RFC 2132 DHCP Options and BOOTP Vendor Extensions.

IMPORTANT:
Option values are returned as they appear in the DHCPACK packet. The component fields of an option value may not be packed as you expect. Check with the RFC before interpreting option values.

The routine expects the option value to fit into the buffer specified by `buf` and `bufSize`. If the buffer is too small to hold the option value, the routine returns `kOTBufferOverflowErr`. If you are requesting all options (`opt` is `kAllDHCPOptions`) and you get this error, the buffer contains the first `bufSize` bytes of the options. If you are requesting a specific option and you get this error, the buffer contains no useful data.

The following snippet of code shows how to extract the subnet mask option from the DHCP packet that was used to configure the computer:

```
enum {
    kDHCPSubnetMaskOption = 1
};

UInt8 optionBuffer[6];

err = OTInetGetDHCPConfigInfo(  (InetDHCPOption*) optionBuffer,
                                sizeof(optionBuffer),
                                kDefaultInetInterface,
                                kDHCPSubnetMaskOption);
if (err == noErr) {
    OTAssert("Got back wrong option", optionBuffer[0] == kDHCPSubnetMaskOption);
    OTAssert("Option size is wrong ", optionBuffer[1] == sizeof(InetHost));
    printf("Subnet mask provided by DHCP is %d.%d.%d.%d\n",
                optionBuffer[2],
                optionBuffer[3],
                optionBuffer[4],
                optionBuffer[5]
            );
}
```

Technical Q&As by API | Number | Technology | Year
Developer Documentation | Technical Notes | Development Kits | Sample Code