

NDIS driver to modify IP headers

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.development.device.drivers/2006-04/msg00488.html>

- *From:* David Sackstein <DavidSackstein@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Tue, 18 Apr 2006 03:58:02 -0700
-

Hi all knowledgeable NDIS miniport driver writers,

My objective is to write a driver that will modify flags in the IP header of certain multicast flows of data that are being sent by a Windows computer. I am using NuMega's DriverNetworks to create the driver and I am able to compile, install and run the driver successfully.

I know that I cannot modify NDIS packets that I did not allocate, so I allocate my own as follows:

```
m_PoolLock.Lock();
KNdisPacket DstPacket = m_TxPool.Allocate();
if (!DstPacket.IsValid())
{
    m_PoolLock.Unlock();
    return NDIS_STATUS_SUCCESS;
}

if (m_BufferIndex+totalLength > BUFFER_SIZE)
m_BufferIndex = 0;

KNdisBuffer Buf = m_BufPool.Allocate( &m_pBuffer[m_BufferIndex]
,totalLength);
m_BufferIndex += totalLength;

DstPacket.ChainAtFront( Buf);
DstPacket.SetFlags(SrcPacket.GetFlags());

NdisMoveMemory(NDIS_OOB_DATA_FROM_PACKET( (PNDIS_PACKET)DstPacket),
NDIS_OOB_DATA_FROM_PACKET( (PNDIS_PACKET)SrcPacket),
sizeof(NDIS_PACKET_OOB_DATA));
DstPacket.STATUS( NDIS_STATUS_PENDING);

// Lock packet buffers before calling NdisCopyFromPacketToPacket (which
// does not according to the DDK's comment). If we fail, complete
// the irp anyway. This would be an indication of low memory conditions.

if (SrcPacket.LockBuffers() && DstPacket.LockBuffers())
{
```

NDIS driver to modify IP headers

```
SrcPacket.CopyFrom(DstPacket, 0, totalLength, 0);  
}  
else  
{  
// TBD handle situation when lock failed – printd, fatal error, etc  
}
```

At the end of this snippet I use Ndis functions to walkthrough and modify the flags I need.

It doesn't work.

That is:

1. I wrote a simple application in C# that sends synthetic data to a multicast address. By using DbgPrint I have asserted that SrcPacket contains the packets with the payload I expect – so the environment of this snippet seems to be functioning.
2. I use Ethereal to view the packets and I do not see any change in the data. I thought this might be because someone is overriding my changes after I make my changes. I know I am processing packets because I traced the payload of the packets from within the processing function in the driver.
3. I switched to another application to send the multicast data. (an application called WinSend that sends MPEG video files). Now I found that I have managed to modify the flags of some packets but not all. Actually, I set
t