# SMB(Server Message Block)
# UDP(User Datagram Protocol)
# ICMP(Internet Control Message Protocol)

기술지원팀

채 희창

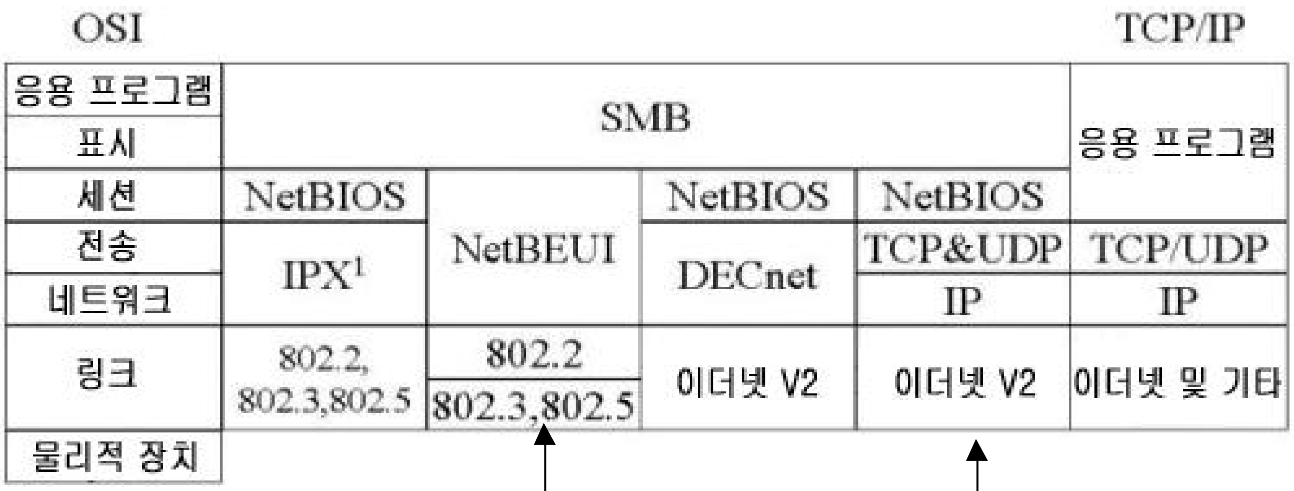
iWORLD (주)아이월드네트워킹

# ✍ SMB (Server Message Block)

## SMB 란?

: Microsoft와 IBM, Intel 에서 공동으로 개발한 프로토콜로 로컬 네트웍에서 파일,프린터 등을 액세스 하는데 사용되는 프로토콜이다. Unix 의 NFS 와 같은 기능을 한다. SMB 는 client/server 모델을 이용한다. Client 는 server에 request (file access,프린팅) 를 보내고 server 는 이 요청을 받아들여서 응답을 보낸다. SMB는 여러 개의 네트웍 프로토콜 상에서 실행될 수 도 있다. 바로 아래층 에는 NetBIOS 프로토콜이 존재한다.

Microsoft 에서는 Window2000 부터 표준 파일 공유 프로토콜로 SMB 를 버전업 시킨 CIFS(Common Internet File System) 1.0 protocol 을 발표 했다.



| OSI | | | | | TCP/IP |
|---|---|---|---|---|---|
| 응용 프로그램 | SMB | | | | 응용 프로그램 |
| 표시 | | | | | |
| 세션 | NetBIOS | NetBEUI | NetBIOS | NetBIOS | |
| 전송 | IPX[1] | | DECnet | TCP&UDP | TCP/UDP |
| 네트워크 | | | | IP | IP |
| 링크 | 802.2,<br>802.3,802.5 | 802.2<br>802.3,802.5 | 이더넷 V2 | 이더넷 V2 | 이더넷 및 기타 |
| 물리적 장치 | | | | | |

많이 쓰임

**NetBIOS over TCP / IP**

# ✍ SMB(Server Message Block)

## ✍ SMB Message-Exchange Sequence

1. SMB_COM_NEGOTIATE
2. SMB_COM_SESSION_SETUP_ANDX
3. SMB_COM_TREE_CONNECT
4. SMB_COM_OPEN
5. SMB_COM_READ
6. SMB_COM_CLOSE
7. SMB_COM_TREE_DISCONNECT

SMB 명령어 set ?

CIFS 는 네트웍상의 다른 컴퓨터에 보내는 요청을 구성한다.

✍ 연결 설정 메시지 - 서버의 공유 리소스에 대한 연결을 시작하고 종료하는 명령으로 구성됩니다.

SMB_COM_NEGOTIATE : 서버와 클라이언트 간에 사용한 명령어 종류

SMB_COM_SESSION_SETUP_ANDX : 사용자 이름, Verification

✍ 이름 공간 및 파일 조작 메시지 - 서버의 파일에 대한 액세스 권한을 얻거나 파일을 읽고 쓰는 데 사용됩니다.

SMB_COM_TREE_CONNECT : client 가 access 하고자 하는 disk 의 이름.

SMB_COM_OPEN,SMB_COM_READ…

✍ 프린터 메시지 - 서버의 인쇄 대기열로 데이터를 보내고 이에 대한 상태 정보를 얻는 데 사용됩니다.

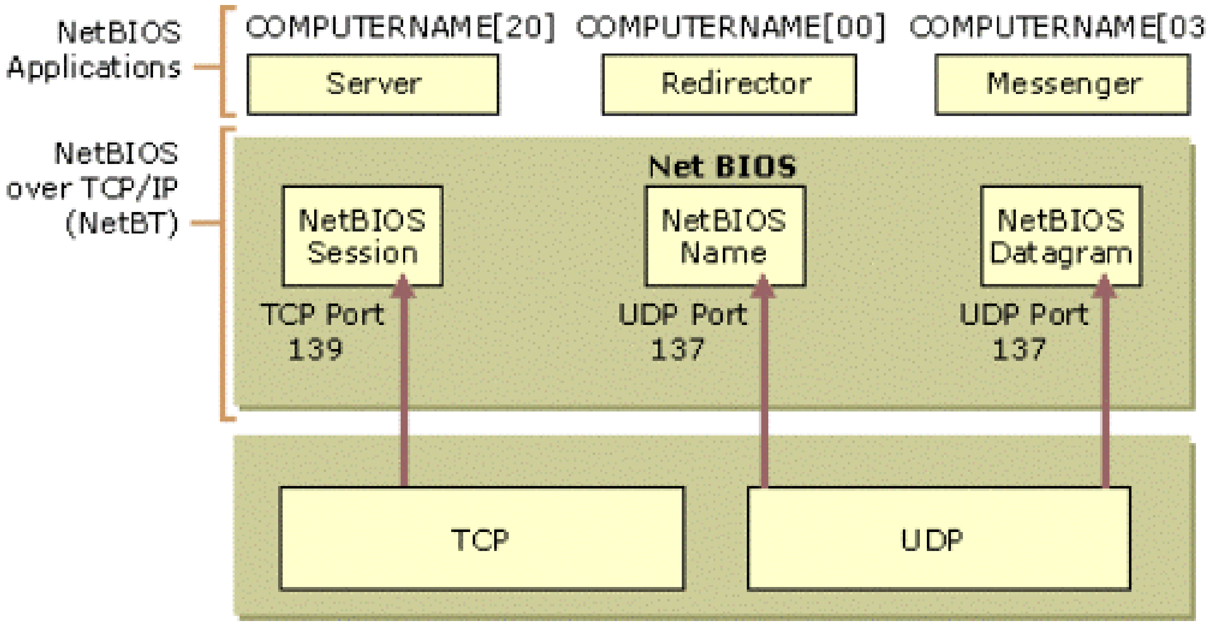
✍ 기타 메시지 - 서버의 메일 슬롯과 명명된 파이프에 쓰는 데 사용됩니다. NETWORK 상에 내가 존재함을 알릴 때

# ✍ SMB (Server Message Block)

NetBIOS 프로토콜 ?

: Windows 네트웍상에서 서버와 클라이언트 상의 application 계층간의 연결 서비스를 제공한다.

여러 개의  NetBIOS 서비스가 있다. 여기서 SMB 와 관련된 서비스는 대표적으로 Server 와 Redirector, Browsing등이 있고 각각의 서비스는 고유한 NetBIOS 이름을 갖는데 이는 컴퓨터이름 (15byte)과 각각의 서비스를 구분 짓는 1byte 로 이루어 진다. 예로 server 서비스는  컴퓨터이름 [20],Browsing 서비스는 WORKGROUP[1D]등등…  여기서 WORKGROUP 은 컴퓨터가 속한 GROUP 이름



**WORKGROUP[1D]**

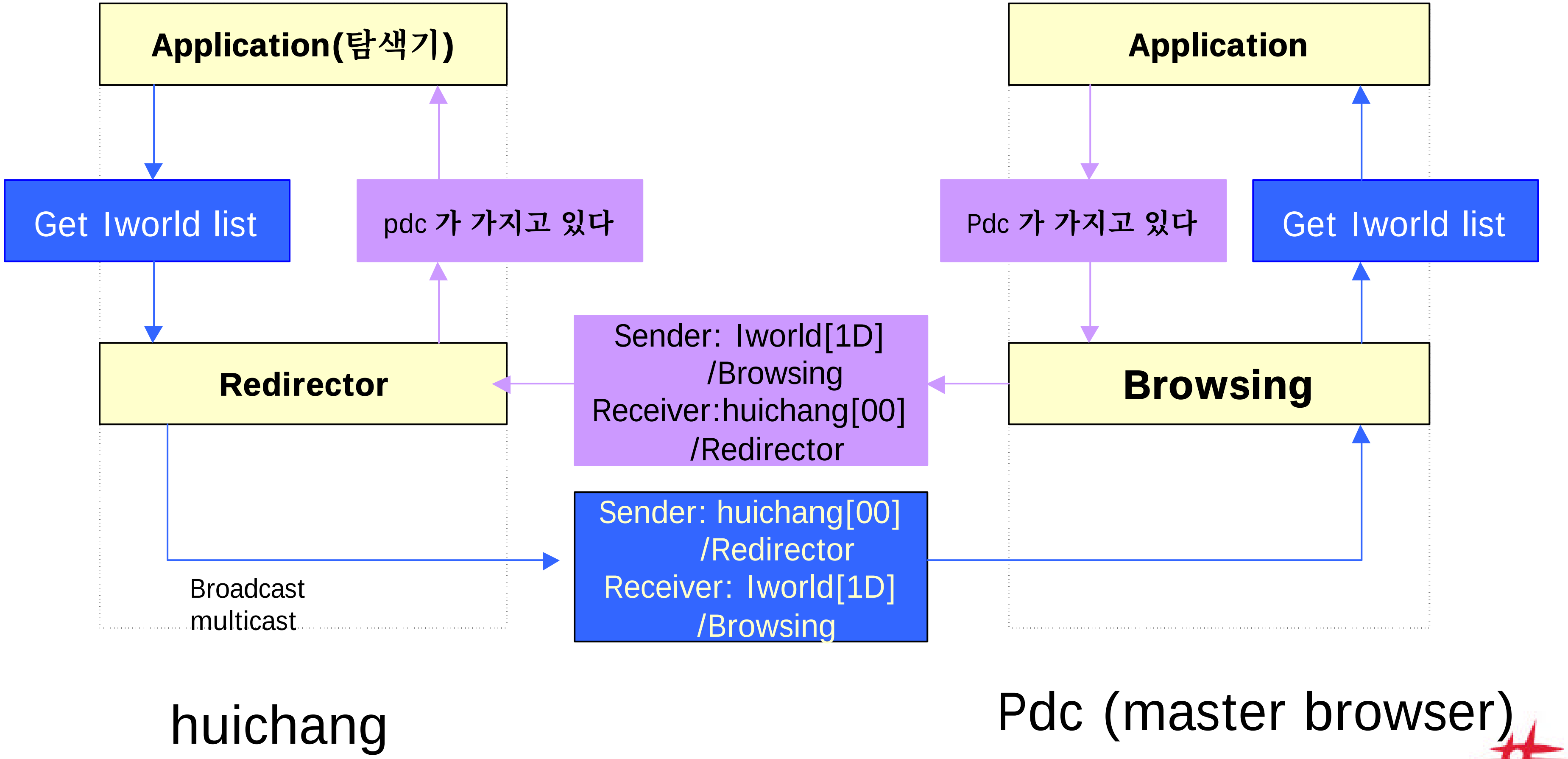
**Browsing**

..

## ✍ SMB (Server Message Block)

클라이언트(huichang)가 Workgroup Iworld 에 속한 컴퓨터 Prosvc 접근하고 자 할 때

1. Iworld 에 속한 컴퓨터의 목록을 요청한다.



Application(탐색기)

Get Iworld list

pdc 가 가지고 있다

Redirector

Sender: Iworld[1D] /Browsing
Receiver:huichang[00] /Redirector

Sender: huichang[00] /Redirector
Receiver: Iworld[1D] /Browsing

Broadcast
multicast

Application

Pdc 가 가지고 있다

Get Iworld list

Browsing

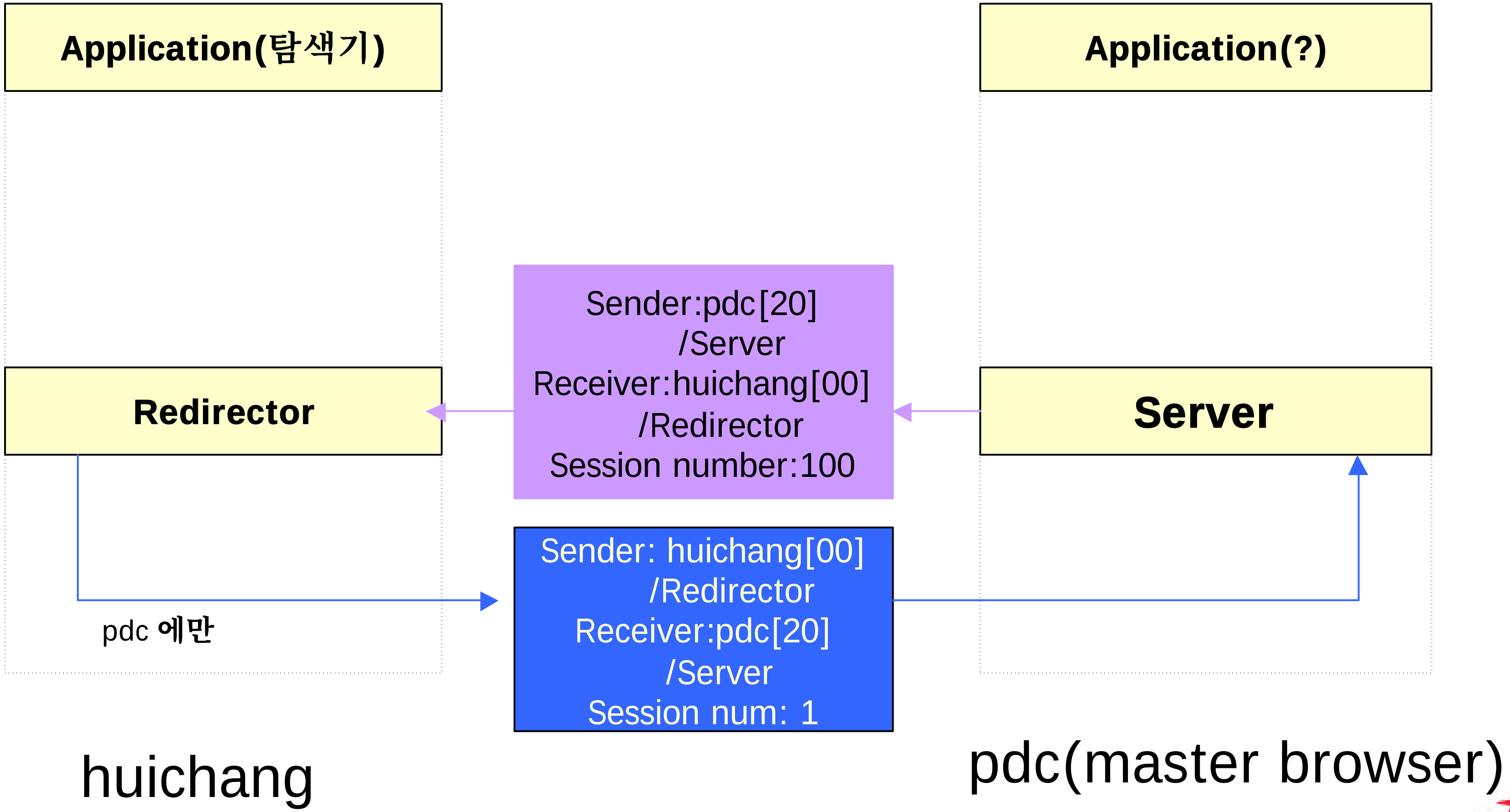
huichang

Pdc (master browser)

# ✍ SMB(Server Message Block)

클라이언트(huichang)가 workgroup Iworld 에 속한 컴퓨터 Prosvc 접근하고 자 할 때

2. Pdc 에 Iworld 에 속한 컴퓨터 목록 요청. Pdc 와 연결을 설정할 때 까지는 NetBIOS 만이 관여.



Application(탐색기)

Application(?)

Sender:pdc[20]
/Server
Receiver:huichang[00]
/Redirector
Session number:100

Redirector

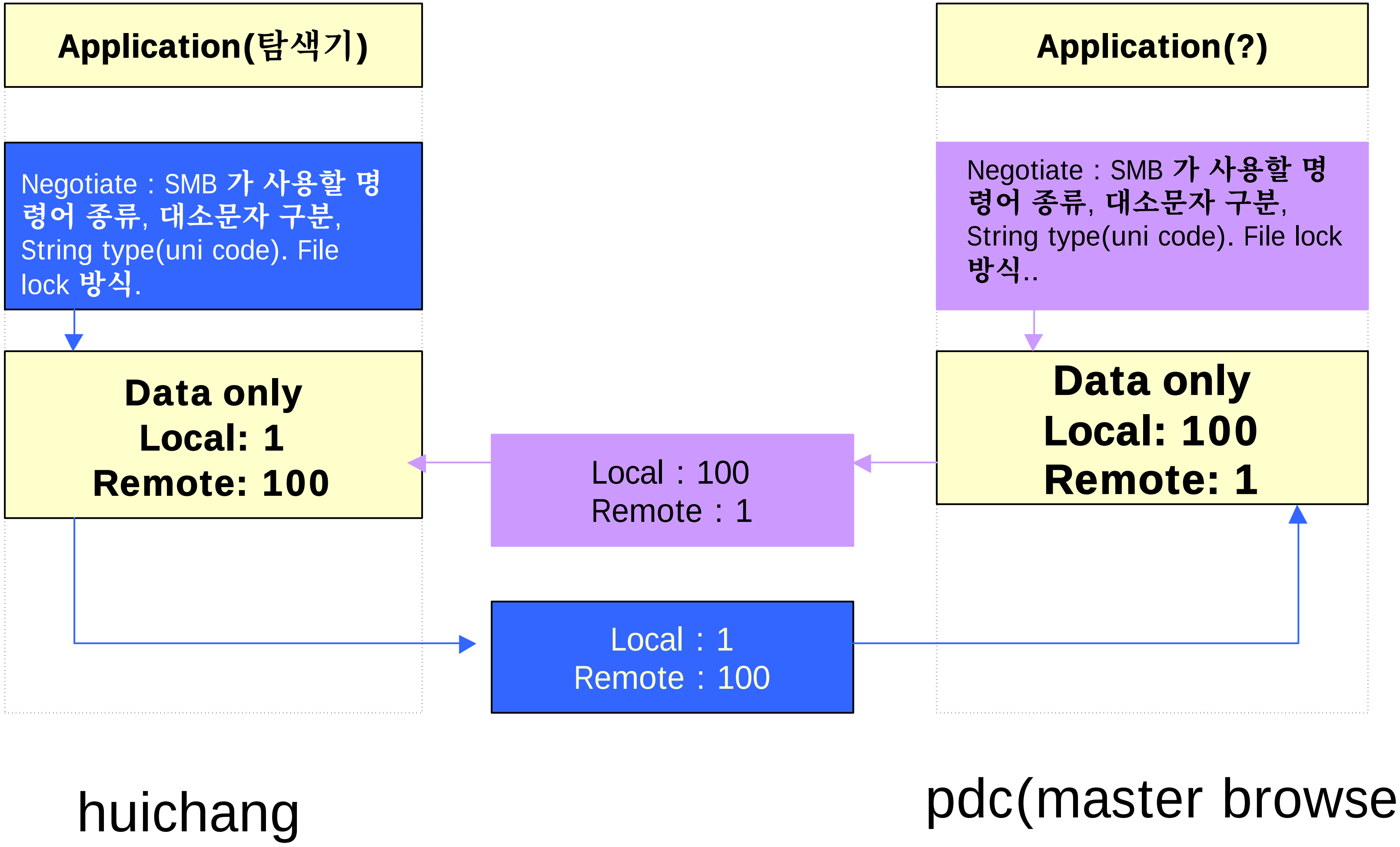
Server

Sender: huichang[00]
/Redirector
Receiver:pdc[20]
/Server
Session num: 1

# ✍ SMB (Server Message Block)

클라이언트(huichang)가 workgroup Iworld 에 속한 컴퓨터 Prosvc 접근하고 자 할 때

3. Pdc 에서 목록을 받아 온다. 여기부터 SMB 프로토콜이 쓰인다.



Application(탐색기)

Application(?)

Negotiate : SMB 가 사용할 명령어 종류, 대소문자 구분, String type(uni code). File lock 방식.

Negotiate : SMB 가 사용할 명령어 종류, 대소문자 구분, String type(uni code). File lock 방식..

Data only
Local: 1
Remote: 100

Local : 100
Remote : 1

Data only
Local: 100
Remote: 1

Local : 1
Remote : 100

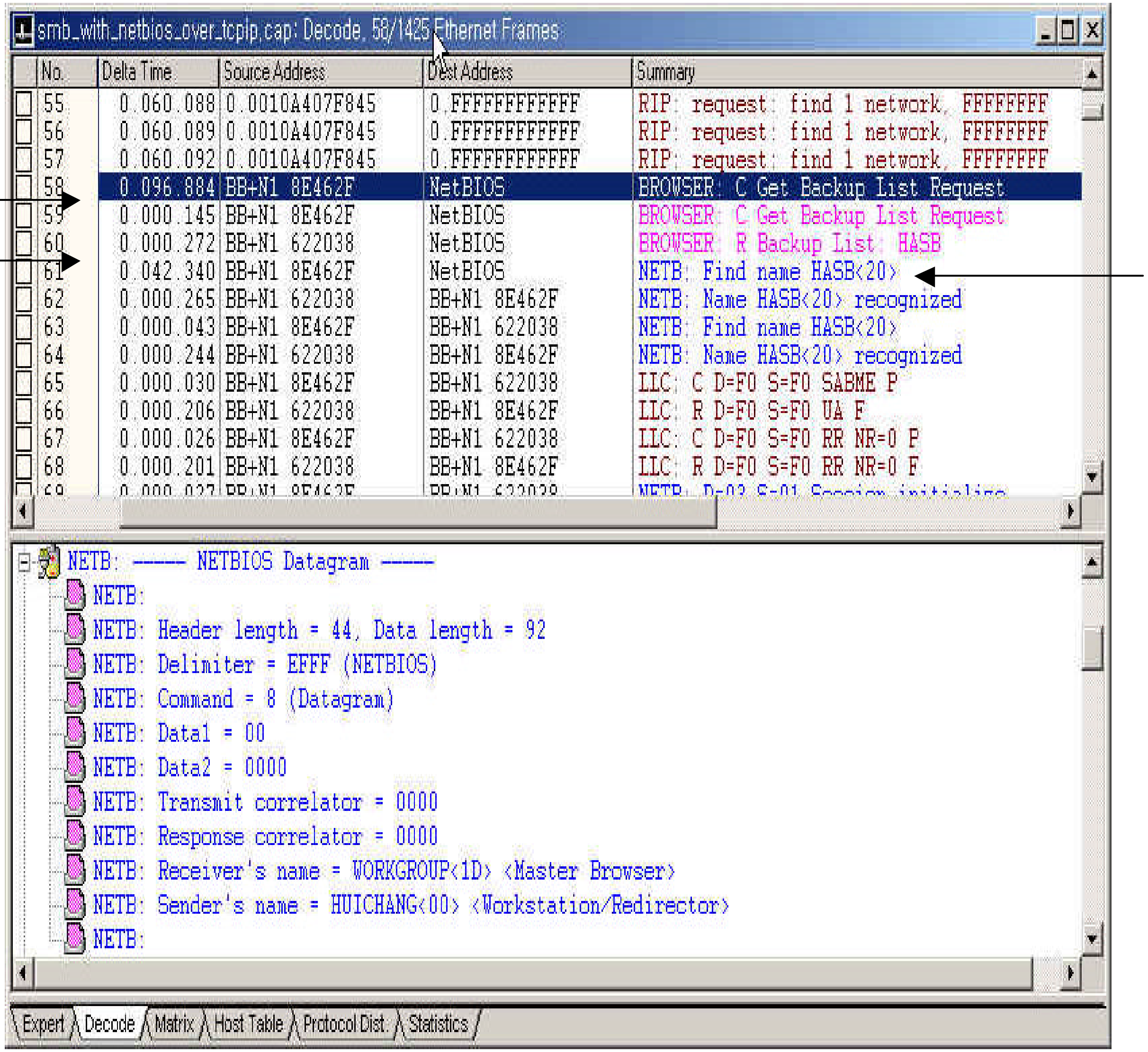
huichang

pdc(master browser)

# ✍ SMB (Server Message Block)

WORKGROUP 에 속한 컴퓨터의 목록을 요청

그 목록은 HSB 가 가지고 있다.



smb_with_netbios_over_tcpip.cap: Decode, 58/1425 Ethernet Frames
Expert
Decode
Matrix
Host Table
Protocol Dist.
Statistics

```
No. Delta Time  Source Address  Dest Address  Summary
55  0.060.088   0.0010A407F845  0.FFFFFFFFFFFF RIP: request: find 1 network, FFFFFFFF
56  0.060.089   0.0010A407F845  0.FFFFFFFFFFFF RIP: request: find 1 network, FFFFFFFF
57  0.060.092   0.0010A407F845  0.FFFFFFFFFFFF RIP: request: find 1 network, FFFFFFFF
58  0.096.884   BB+N1 8E462F    NetBIOS        BROWSER: C Get Backup List Request
59  0.000.145   BB+N1 8E462F    NetBIOS        BROWSER: C Get Backup List Request
60  0.000.272   BB+N1 622038    NetBIOS        BROWSER: R Backup List: HASB
61  0.042.340   BB+N1 8E462F    NetBIOS        NETB: Find name HASB<20>
62  0.000.265   BB+N1 622038    BB+N1 8E462F   NETB: Name HASB<20> recognized
63  0.000.043   BB+N1 8E462F    BB+N1 622038   NETB: Find name HASB<20>
64  0.000.244   BB+N1 622038    BB+N1 8E462F   NETB: Name HASB<20> recognized
65  0.000.030   BB+N1 8E462F    BB+N1 622038   LLC: C D=F0 S=F0 SABME P
66  0.000.206   BB+N1 622038    BB+N1 8E462F   LLC: R D=F0 S=F0 UA F
67  0.000.026   BB+N1 8E462F    BB+N1 622038   LLC: C D=F0 S=F0 RR NR=0 P
68  0.000.201   BB+N1 622038    BB+N1 8E462F   LLC: R D=F0 S=F0 RR NR=0 F
69  0.000.027   BB+N1 8E462F    BB+N1 622038   NETB: D=03 S=01 Session initialize
```

```
NETB: ----- NETBIOS Datagram -----
    NETB:
    NETB: Header length = 44, Data length = 92
    NETB: Delimiter = EFFF (NETBIOS)
    NETB: Command = 8 (Datagram)
    NETB: Data1 = 00
    NETB: Data2 = 0000
    NETB: Transmit correlator = 0000
    NETB: Response correlator = 0000
    NETB: Receiver's name = WORKGROUP<1D> <Master Browser>
    NETB: Sender's name = HUICHANG<00> <Workstation/Redirector>
    NETB:
```

HASB 에 연결

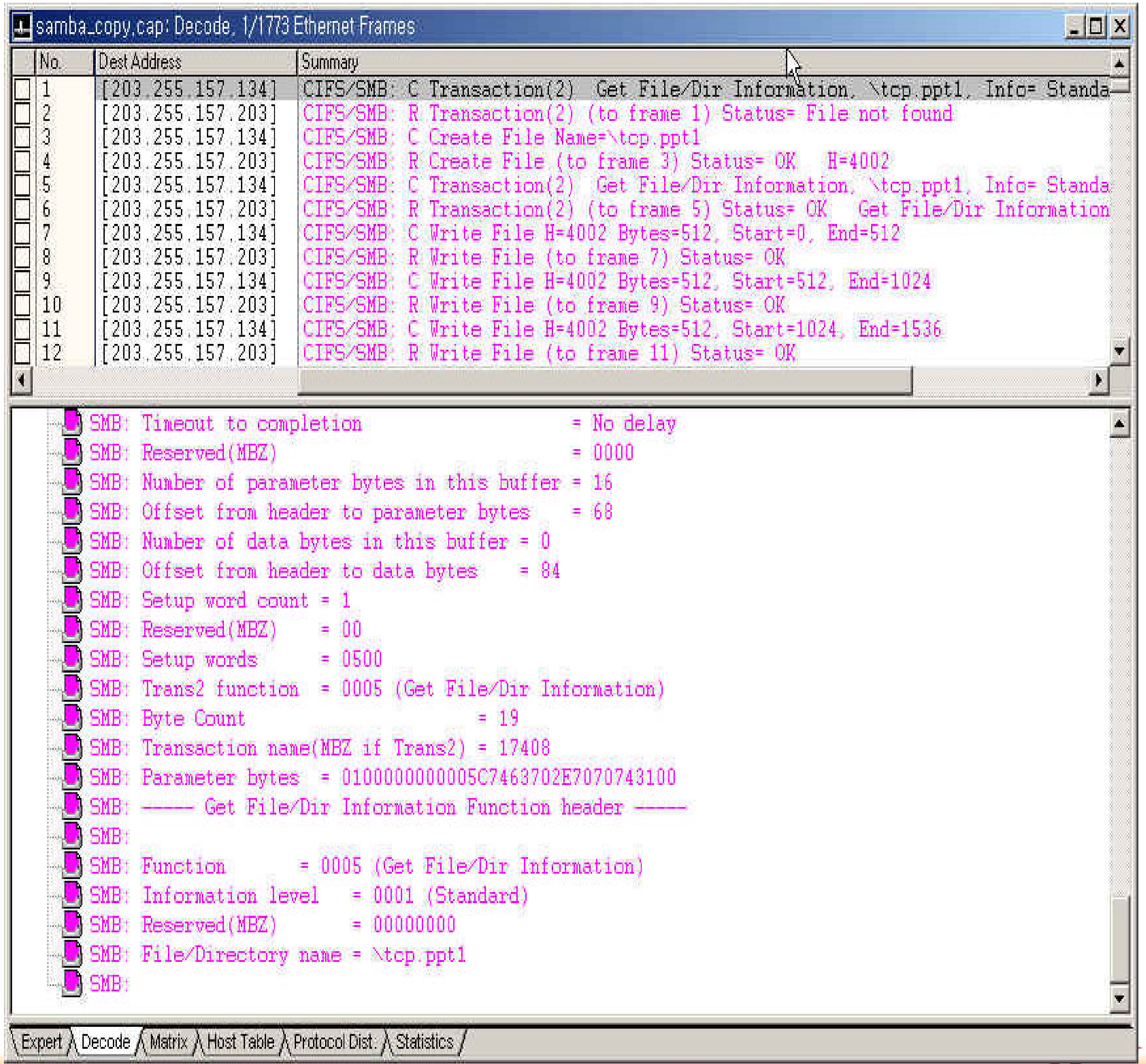
NetBIOS 연결.

# ✍ SMB (Server Message Block)

옆의 그림은 tcp.ppt1 이라는 파일을 203 에서 134 으로 복사하는 경우에 SMB 의 행동을 보여 주고 있다.

먼저 203 에서는134에tcp.ppt1이라는 파일이 있는지를 확인한다. 응답은 없다. 그러면 203은 134 에 파일 tcp.ppt1 을 생성하라고 명령. 응답은 생성했음. 그러면 203은 134에 tcp.ppt1에 대한 정보를 요구. 응답은 Tcp.ppt1 에 대한 정보.

이후에 203은 SMB 에 data의 양과 함께 파일에 쓰라는 명령을 보냄.



samba_copy.cap: Decode, 1/1773 Ethernet Frames

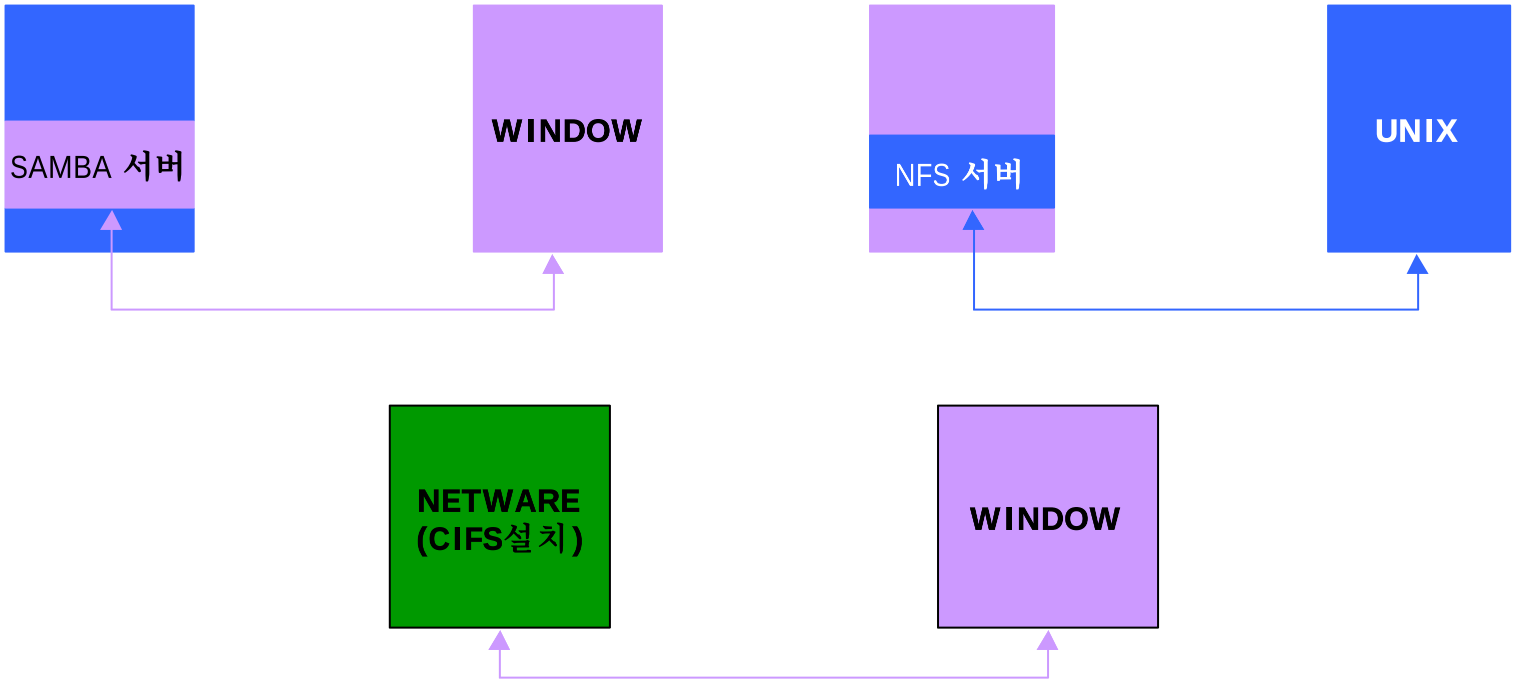
```
No.  Dest Address        Summary
1    [203.255.157.134]   CIFS/SMB: C Transaction(2)  Get File/Dir Information, \tcp.ppt1, Info= Standa
2    [203.255.157.203]   CIFS/SMB: R Transaction(2) (to frame 1) Status= File not found
3    [203.255.157.134]   CIFS/SMB: C Create File Name=\tcp.ppt1
4    [203.255.157.203]   CIFS/SMB: R Create File (to frame 3) Status= OK   H=4002
5    [203.255.157.134]   CIFS/SMB: C Transaction(2)  Get File/Dir Information, \tcp.ppt1, Info= Standa
6    [203.255.157.203]   CIFS/SMB: R Transaction(2) (to frame 5) Status= OK   Get File/Dir Information
7    [203.255.157.134]   CIFS/SMB: C Write File H=4002 Bytes=512, Start=0, End=512
8    [203.255.157.203]   CIFS/SMB: R Write File (to frame 7) Status= OK
9    [203.255.157.134]   CIFS/SMB: C Write File H=4002 Bytes=512, Start=512, End=1024
10   [203.255.157.203]   CIFS/SMB: R Write File (to frame 9) Status= OK
11   [203.255.157.134]   CIFS/SMB: C Write File H=4002 Bytes=512, Start=1024, End=1536
12   [203.255.157.203]   CIFS/SMB: R Write File (to frame 11) Status= OK
```

```
SMB: Timeout to completion                        = No delay
SMB: Reserved(MBZ)                                = 0000
SMB: Number of parameter bytes in this buffer = 16
SMB: Offset from header to parameter bytes        = 68
SMB: Number of data bytes in this buffer = 0
SMB: Offset from header to data bytes      = 84
SMB: Setup word count = 1
SMB: Reserved(MBZ)     = 00
SMB: Setup words       = 0500
SMB: Trans2 function   = 0005 (Get File/Dir Information)
SMB: Byte Count                              = 19
SMB: Transaction name(MBZ if Trans2) = 17408
SMB: Parameter bytes   = 0100000000005C7463702E7070743100
SMB: ----- Get File/Dir Information Function header -----
SMB:
SMB: Function          = 0005 (Get File/Dir Information)
SMB: Information level    = 0001 (Standard)
SMB: Reserved(MBZ)        = 00000000
SMB: File/Directory name = \tcp.ppt1
SMB:
```

# ✍ SMB (Server Message Block)

SMB 는 Window 계열의 컴퓨터에서 자원을 공유하기 위한 프로토콜로 설계되었다. 유닉스 계열에서는 이미 NFS 같은 네트웍공유 프로토콜이 존재하고 있었다. 이후에 Window 와 유닉스 간의 자원 공유를 위하여 SAMBA(유닉스용) 나 Window 용의 NFS 서버나 클라이언트가 생겨났다.

Novell NetWare 서버에도 CIFS 추가기능 설치하면 Windows 2000 서버처럼 Windows 기반 클라이언트에 응답한다. 이외에도 Windows 2000 이 Netware 서버에 대한 Gateway 역할을 할 수도 있다.



# ✍ UDP(User Datagram Protocol) & ICMP(Internet Control Message Protocol)

: UDP 는 사용자 process가 write 한 만큼의 정보로 하나의 datagram 을 형성한다. 하나의 datagram 의 최대크기는 65535 byte(Ip header length field 가 16 bit) 까지 이다. 헤더는 간단하게 8 byte 로 이루어져 있다. Source,destination port number,length field checksum 으로 이루어져 있다. TCP 와 같은 데이터의 완전한 전송을 보장하기 위한 어떠한 field 도 존재하지 않는다. 따라서 UDP 는 동영상 등의 중요하지 않은 정보 전달에 이용된다.

| | |
|---|---|
| 16-bit source port num | 16-bit dest port num |
| 16-bit UDP length | 16-bit UDP checksum |
| Data (사용자 프로세스가 write 한) | |

# ✍ UDP(User Datagram Protocol) & ICMP(Internet Control Message Protocol)

## ✍ UDP 에서 부족한 점?

1. Flow control : 받는 쪽에서 받은 데이터를 모두 프로세스에 전달하는가 ?
2. Fragmentation : 프로세스가 쓴 만큼의 데이터가 하나의 datagram 을 형성하여 fragmentation 의 가능성.
3. Reliability : 데이터가 제대로 전달 되는가 ? - 목적지 호스트가 여전히 데이터를 받고 있는가?

## ✍ UDP 의 부족한 점을 보완 하기 위한 방법 – ICMP protocol 이용

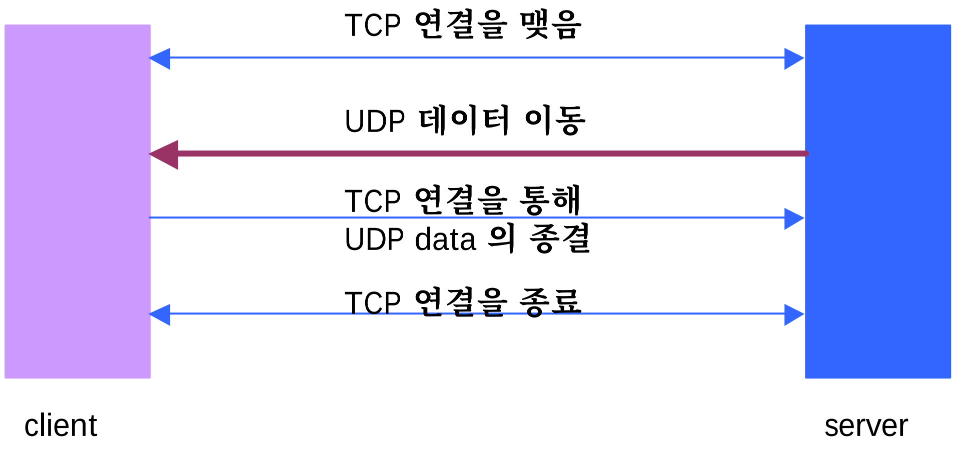
Flow control – 받는 쪽에서 데이터가 버퍼에서 버려 질 때 받는 host 는 icmp source quench error (type 4 , code 0) 발생. 보내는 쪽에서는 이 icmp 메시지를 이용할 수 있다.

Fragmentation – 보내는 쪽에서 UDP 데이터그램의 IP 헤더의 DF 을 set 하여 보내면 중간에 fragment 가 필요한 부분에서 icmp need to fragment(type 3, code 4) 메시지를 받는다.

Reliability - UDP 가 존재하지 않는 host 에 데이트를 보내게 되면 gateway 는 icmp host unreachable (type 3, code 1) 메시지 발생. 혹은 host 는 존재하나 받는 프로세스가 없을 때는 icmp protocol unreachable(type 3, code 3) 메시지 발생.

# ✍ UDP(User Datagram Protocol) & ICMP(Internet Control Message Protocol)

✍ UDP 를 이용한 동영상 프로그램의 작동

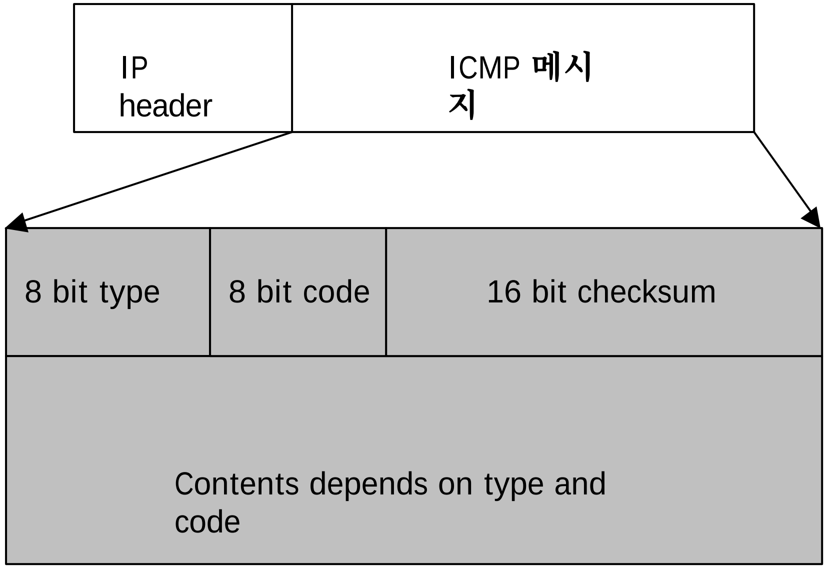


UDP 를 이용한 동영상 전송 프로그램은 ICMP 의 메시지를 이용하지 않는다. 대신에 TCP 연결을 통해서 연결과 종료의 메시지를 주고 받는다.

# ✍ UDP(User Datagram Protocol) & ICMP(Internet Control Message Protocol)

✍ ICMP ?

: IP 계층에서 작동하는 프로토콜로서 error 메시지나 주의(attention) 가 필요한 상태를 나타내는데 사용된다.



IP header

ICMP 메시지

| 8 bit type | 8 bit code | 16 bit checksum |
|---|---|---|

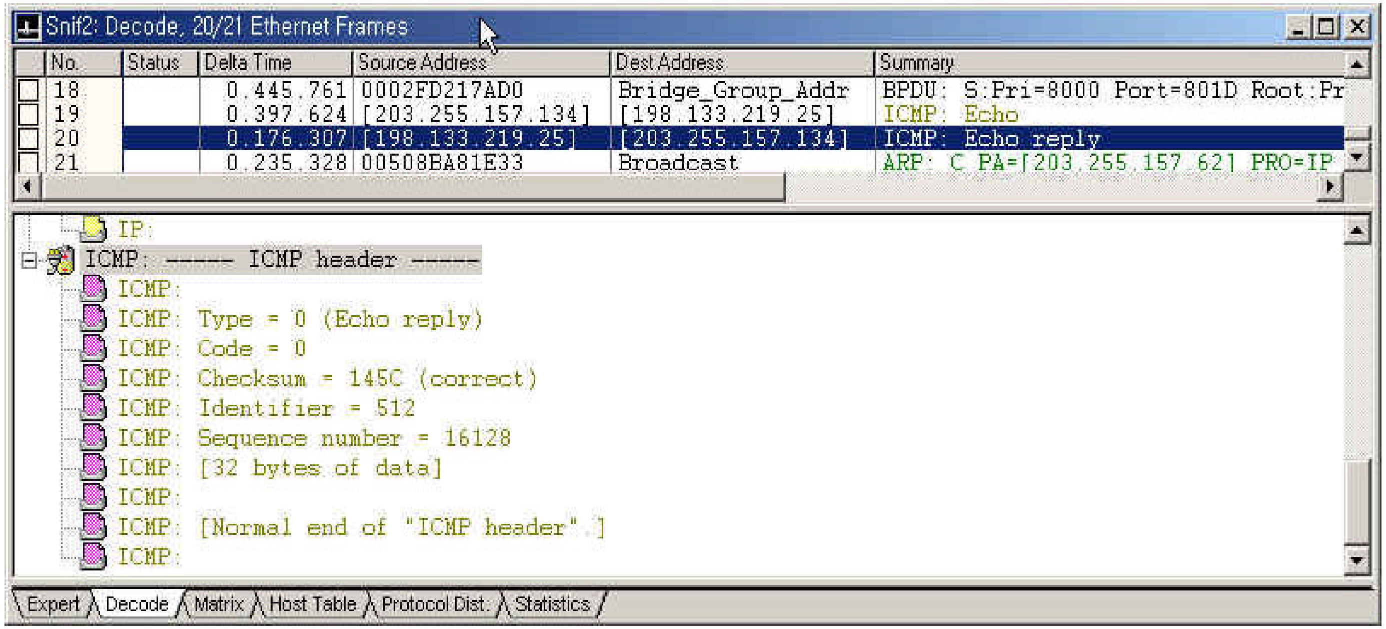
Contents depends on type and code

| | | type | code |
|---|---|---|---|
| Echo request | : | 8 | 0 |
| Echo reply | : | 0 | 0 |
| Source quench | : | 4 | 0 |
| Network unreachable | : | 3 | 0 |
| Host unreachable | : | 3 | 1 |
| Protocol unreachable | : | 3 | 2 |
| Port unreachable | : | 3 | 3 |
| Fragment needed(DF bit set) | : | 3 | 4 |
| Time to live equals 0 | : | 11 | 0 |

# ✍ UDP(User Datagram Protocol) & ICMP(Internet Control Message Protocol)

- ✍ ICMP echo request and reply

<table>
<tr><td>8 bit type</td><td>8 bit code</td><td>16 bit checksum</td></tr>
<tr><td colspan="2">16 bit identifier</td><td>16 bit sequence number</td></tr>
<tr><td colspan="3">Contents depends on type and code</td></tr>
</table>



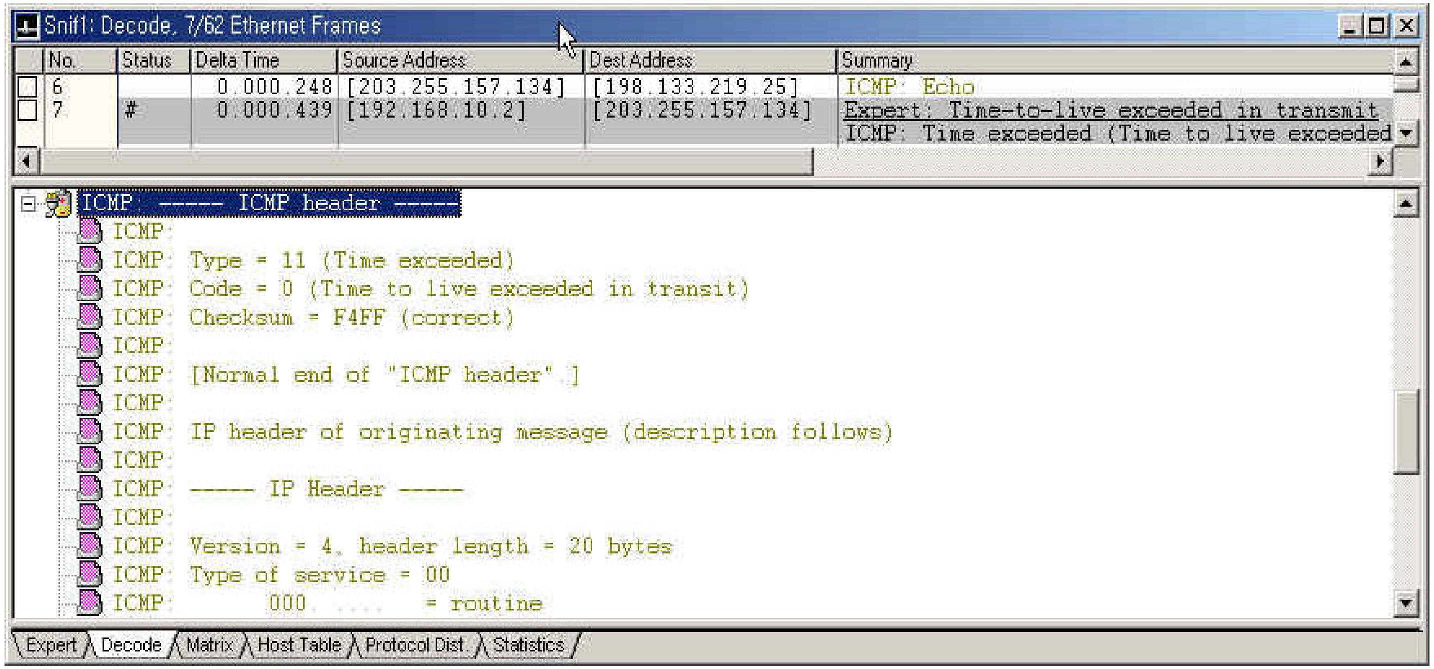
| No. | Status | Delta Time | Source Address | Dest Address | Summary |
|---|---|---|---|---|---|
| 18 | | 0.445.761 | 0002FD217AD0 | Bridge_Group_Addr | BPDU: S:Pri=8000 Port=801D Root:Pr |
| 19 | | 0.397.624 | [203.255.157.134] | [198.133.219.25] | ICMP: Echo |
| 20 | | 0.176.307 | [198.133.219.25] | [203.255.157.134] | ICMP: Echo reply |
| 21 | | 0.235.328 | 00508BA81E33 | Broadcast | ARP: C PA=[203.255.157.62] PRO=IP |

```
  IP:
⊟ ICMP: ----- ICMP header -----
    ICMP:
    ICMP: Type = 0 (Echo reply)
    ICMP: Code = 0
    ICMP: Checksum = 145C (correct)
    ICMP: Identifier = 512
    ICMP: Sequence number = 16128
    ICMP: [32 bytes of data]
    ICMP:
    ICMP: [Normal end of "ICMP header".]
    ICMP:
```

# ✍ UDP(User Datagram Protocol) & ICMP(Internet Control Message Protocol)

✍ ICMP time to live equals 0

| 8 bit type | 8 bit code | 16 bit checksum |
|---|---|---|
| Contents depends on type and code | | |



Snif1: Decode, 7/62 Ethernet Frames

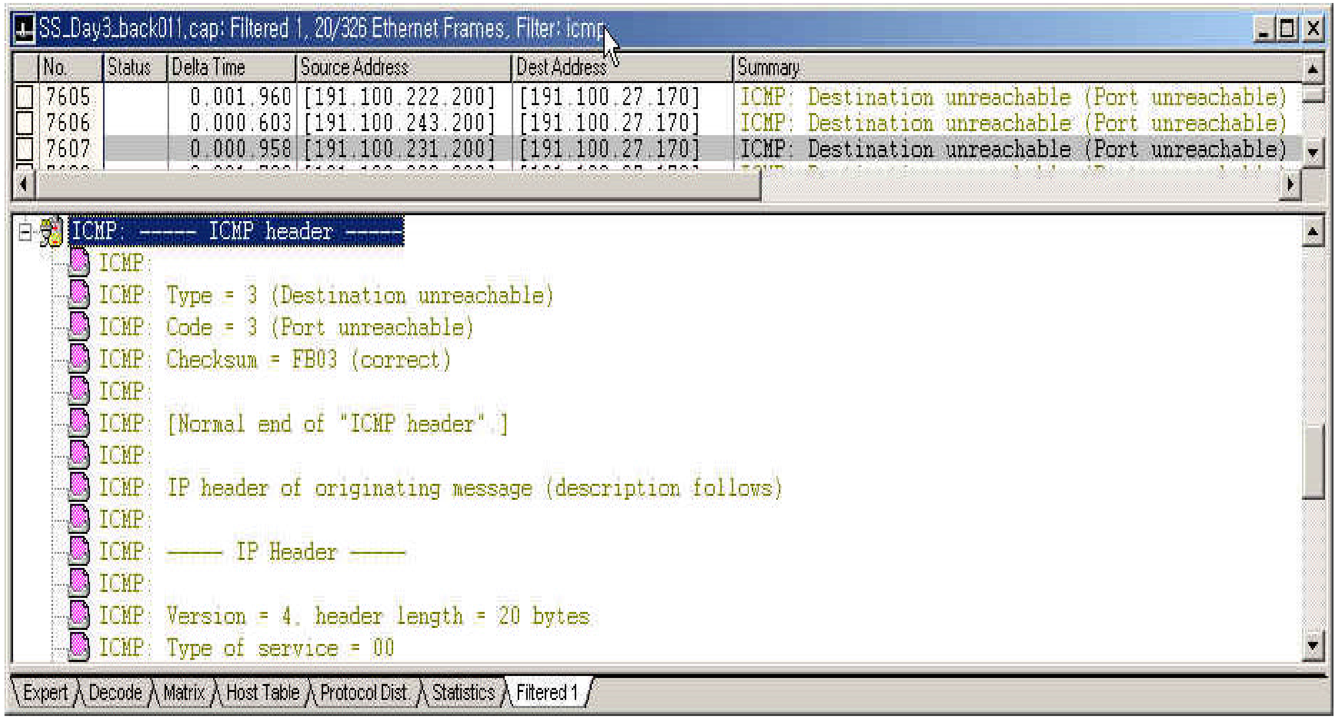
```
No.  Status  Delta Time  Source Address     Dest Address       Summary
6             0.000.248  [203.255.157.134]  [198.133.219.25]   ICMP: Echo
7    #        0.000.439  [192.168.10.2]     [203.255.157.134]  Expert: Time-to-live exceeded in transmit
                                                                ICMP: Time exceeded (Time to live exceeded
```

```
ICMP: ----- ICMP header -----
  ICMP:
  ICMP: Type = 11 (Time exceeded)
  ICMP: Code = 0 (Time to live exceeded in transit)
  ICMP: Checksum = F4FF (correct)
  ICMP:
  ICMP: [Normal end of "ICMP header".]
  ICMP:
  ICMP: IP header of originating message (description follows)
  ICMP:
  ICMP: ----- IP Header -----
  ICMP:
  ICMP: Version = 4, header length = 20 bytes
  ICMP: Type of service = 00
  ICMP:       000. ....  = routine
```

# ✍ UDP(User Datagram Protocol) & ICMP(Internet Control Message Protocol)

- ICMP port unreachable

| 8 bit type | 8 bit code | 16 bit checksum |
|---|---|---|
| Contents depends on type and code | | |



SS_Day3_back011.cap: Filtered 1, 20/326 Ethernet Frames, Filter: icmp

```
No.   Status  Delta Time   Source Address      Dest Address       Summary
7605          0.001.960    [191.100.222.200]   [191.100.27.170]   ICMP: Destination unreachable (Port unreachable)
7606          0.000.603    [191.100.243.200]   [191.100.27.170]   ICMP: Destination unreachable (Port unreachable)
7607          0.000.958    [191.100.231.200]   [191.100.27.170]   ICMP: Destination unreachable (Port unreachable)
```

```
ICMP: ----- ICMP header -----
ICMP:
ICMP: Type = 3 (Destination unreachable)
ICMP: Code = 3 (Port unreachable)
ICMP: Checksum = FB03 (correct)
ICMP:
ICMP: [Normal end of "ICMP header".]
ICMP:
ICMP: IP header of originating message (description follows)
ICMP:
ICMP: ----- IP Header -----
ICMP:
ICMP: Version = 4, header length = 20 bytes
ICMP: Type of service = 00
```