TCP/IP over ATM (Asynchronous Transfer Mode)

ATM Background

Based in telco technology
Analog −−> ISDN (STM) −−> BISDN(ATM)

Motivation

Telco’s were *early* users of digital (vs. analog) transmission
(but its easy to get locked into a mindset that is not advantageous for the long term!)

Long distance trunks multiplexed PCM (synchronous − 1960’s) channels

ISDN (It Still Does Nothing) provided these digital channels to end users in the 1980’s

Disadvantages of ISDN:

Not possible to take advantage of audio compression
(which was not (economically) feasible when PCM was developed).

The 64Kb synchronous channel is wasteful (and costly) for bursty data circuits.

Objectives of ATM:

Provide QoS guarantees to multiple traffic classes
CBR (still stuck in PCM mindset).
Real time VBR traffic (compressed video).
Non−realtime VBR traffic (traditional packet data).

Support *very* high switching rates
Giga cells / second
=> No 10,000 entry table lookups

Avoid "Head of Line" blocking problems of packet switched nets.

Services provided by ATM:

Connection oriented
Unreliable (packet loss and corruption possible)
QoS guaranteed.

Mechanism

Hierarchical connection orientation (vpi, vci) with connection id based switching

=> No 10,000+ entry table lookup
12 bit index makes hardware lookup possible in routers.

Connection identifier (vpi, vci) is
dynamic (changes from hop to hop)
independent of the ATM address of either end point

ATM addresses vs. IP addresses

In IP we saw that two distinct address classes were used
IP addresses routed the packet to the target LAN
The ARP mechanism mapped IP to MAC level addresses for final delivery

In ATM the "global" and "local" parts are combined into a 20 byte! internationally unique addresses.

ATM addresses come in three formats

DCC (0x39)(Domain country code)
AFI DCC HO–DSP ESI SEL
ICD (0x47)(International code designator)
AFI ICD HO–DSP ESI SEL
E.164 (0x45)(Telephone number)
AFI E.164 HO–DSP ESI SEL

AFI = Address format identifier (1 byte)
DCC = Domain country code (2 bytes)
ICD = International code designator (2 bytes)
E.164 = Phone number 10 BCD bytes
HO–DSP = High order domain specific (like an IP (V6) address)
ESI = End System Identifier (6 bytes like a MAC address)
SEL = Selector (Allows an endsystem to support 255 connections (1 Byte))

```
47.0203.04050607080900000104.0004AC6C28F1.00 atm-atm
47.0203.04050607080900000104.0004AC6C308A.00 chat-atm
47.0203.04050607080900000101.111111111111.11 arpsrv-atm
47.0203.04050607080900000201.999999999999.00 fd8285-atm
47.0203.04050607080900000201.0004AC6C2B7F.00 homer-atm
47.0203.04050607080900000201.0004ac6c28ef.00 bart-atm
```

Bottom Line

ATM equipped end systems should be able to connect worldwide using ATM enabled routers

IP should just quietly *go away* (but a TCP derivative is needed to provide a reliable transport.)

Problem: Haven't we heard this (X.25 ISDN) before.

The *REAL* obstacle

Massive installed application base

=> eliminating TCP/IP is just about as difficult (but not nearly so beneficial) as eliminating the M$/W*ndows family of Operating Systems

The "Interim" solution

Use ATM as a "virtual" link level protcol connecting IP routers

There are two levels at which this is being done

> Transport backbones (e.g. the vBNS)
> Local LAN's

The main (unresolved) problems with doing so

> Preserving ATM's QoS guarantee's across IP routers
> Provide IP's (allegedly) efficient multicast capability

Low level ATM details

The basic unit of transfer is the *cell*
Cells are fixed in size at 53 bytes and have a 5 byte header

Header fields:

UNI		NNI
GFC	4	
VPI	8	12
VCI	16	16
PTI	3	3
CLP	1	1
HEC	8	8

The cell is routed via (vpi, vci) fields
NNI routing is vpi based
UNI routing is (vpi, vci) based
vpi and vci are *switch* assigned and may vary long the path.

"Real" routing must be done during circuit setup..

There exist protocols for routing in both public and private networks
The standard for private networks is called PNNI

ATM Adaptation layers

Transport like protocols used to *adapt* the raw cell pipe provided by ATM to needs of a particular application

AAL1 – CBR
AAL2 – RT/VBR (But hopelessly broken and since withdrawn)
AAL3/4 Connection oriented/connection less NRT/VBR
––––––––––––––––––––––––––––––––––––
AAL5 – Connection oriented VBR

In practice only AAL1 and AAL5 are of interest in computer networks
AAL5 is used for IP over ATM

AAL5

Frames are up to 65535 bytes long

Payload | CS Trailer

Trailer contains

UU	1
Length	2
CRC	4

Services

Reliable connection oriented with flow control

Unreliable connection oriented

Implementation

AAL’s are fully implemented in adapter microcode

Your AAL5 may not give you reliable service.

ATM Connections

PVC’s: permanent virtual circuits

Like a leased line

Setup by system administrator

Admin: Your connection to host jmw is (vpi, vci)

SVC’s: switched virtual circuits

Like a telephone call

Connection is made by sending ATM address on PVC (0,5)

Connection persists until manually taken down

```
End system:                                   Switch
          ------- Contact "20 byte address"  --->
         <-------- Your connection id is (vpi,vci)
```

Classical IP over ATM (CLIP / CIP)

Important RFC's

RFC 1577 Classical IP and ARP over ATM
RFC 1626 Default IP MTU for IP over ATM with AAL5
RFC 1755 ATM Signaling support for IP over ATM
RFC 1932 Comparison of various models for IP over ATM.

CLIP is based on the Logical IP Subnet (LIS)

Each LIS must have its own network/subnet address
Each LIS must be connected to the rest of the internet though an IP router
The Router must have in an interface on the LIS.
All elements of the LIS must belong to the same ATM network
  (it must be possible to establish at ATM SVC or PVC between them)
Any element that is NOT on the ATM network cannot belong to the LIS
  (modulo the gateway router)
Each element must know via ARP how to contact every other element.

Encapsulation of PDUs

Two basic approaches
  Use IEEE 802.2 LLC encapsulation to provide protocol ID
    LLC/SNAP header is used
  Use VCC based multiplexing... dedicate a channel for each network protocol

Classical IP over PVC's

Sys admin configures a full mesh of vcc's
Typical mechanism: setup_vcc (vpi, vci) port port
For example one switch 8 stations:
  need 32 (vpi, vci's)
Must be unique within switch
  within constraints of the workstation (e.g. vpi = 0; vci = 1,..63)
  –– not too hard here
Problem: add another switch with 8 attached workstations

Mapping between VC's and IP addresses

In linux this is done by the combination of the atmarp server and atmarp daemon

Configuring CLIP over PVC's on Linux

Start the arp daemon
–b => background
–d => create debugging log
Create the interface
Configure the interface
Define routing
Specify binding of remote IP/PVC

```
#Run this script on host 1
IPADDR="192.168.1.1"
DEST_IPADDR="192.168.1.2"
NET_IPADDR="192.168.1.0"
PVC="0.0.32"


/usr/local/sbin/atmarpd -b -d >> startip.log 2>&1
/usr/local/sbin/atmarp -c atm0 >> startip.log 2>&1
/sbin/ifconfig atm0 $IPADDR up mtu 2000  >> startip.log 2>&1
/sbin/route add -net $NET_IPADDR netmask 255.255.255.0 dev atm0
/usr/local/sbin/atmarp -s $DEST_IPADDR $PVC >> startip.log 2>&1
```

```
# Run this script on host 2
IPADDR="192.168.1.2"
DEST_IPADDR="192.168.1.1"
NET_IPADDR="192.168.1.0"
PVC="0.0.32"

/usr/local/sbin/atmarpd -b -d >> startip.log 2>&1
/usr/local/sbin/atmarp -c atm0 >> startip.log 2>&1
/sbin/ifconfig atm0 $IPADDR up >> startip.log 2>&1
/sbin/route add -net $NET_IPADDR netmask 255.255.255.0 dev atm0
/usr/local/sbin/atmarp -s $DEST_IPADDR $PVC >> startip.log 2>&1
```

CLIP over SVC’s

One machine on the LIS is designated *arpserver*
Each machine that comes up "calls" the arp server and registers its ATM and IP addresses
ARP server entries expire after 20 minutes.
At that point the server revalidates with InATMARP (like RARP) or discards

Client entries expire after 15 minutes but may also be revalidate with InATMARP
A single VCC may (or may not) support multiple TCP sessions

```
#!/bin/sh
#
# atm.init
#
#       Bring up/down Classical IP over ATM using SVCs and signalling
#

# Notes:
# You must be superuser to execute this script.
# Run this script first on the host that is the arp server, then the others.
# In our case, atm is the arp server.
# mtu is set to 2000

# Edit for your configuration
NET_IPADDR="192.168.1.0"
IPADDR_ATM="192.168.1.2"
IPADDR_CHA="192.168.1.1"
ATMADDR_ATM="39.99.99.99.99.99.99.00.00.99.99.01.01.18.99.99.99.99.01.00"
ATMADDR_CHA="39.99.99.99.99.99.99.00.00.99.99.01.01.18.99.99.99.99.02.00"
#LOGPATH="/var/adm/atm"
LOGPATH="/h10/var/adm/atm"
MTU=2000

    echo "Starting signalization daemon"
    /usr/local/sbin/atmsigd -b
    sleep 2

    echo "Starting ilmi daemon"
    /usr/local/sbin/ilmid -b
    sleep 2

    echo "Starting atmarp daemon"
    /usr/local/sbin/atmarpd -b
    sleep 3

    echo "Creating atm interfaces"
    /usr/local/sbin/atmarp -c atm0
    /sbin/ifconfig atm0 $IPADDR_ATM up mtu $MTU
    /sbin/route add -net $NET_IPADDR netmask 255.255.255.0 dev atm0

    # The following was advised but does not seem to make a difference
    /usr/local/sbin/atmarp -q $NET_IPADDR ubr:sdu=$MTU
```

On the system that doesn’t run the arp server we have to say where to find it.

```
    echo "Create an ARP server entry in local arp table
    /usr/local/sbin/atmarp -s $IPADDR_ATM $ATMADDR_ATM temp arpsrv
```

Lane Emulation (LANE) (ATM Forum)

ATM network emulates

Ethernet
Token Ring (IBM a major proponent)

Bridged ELAN can contain
ATM switches
ATM hosts
LANE bridges
Ethernet segments

Components

LECs	Client workstations (end systems)
LECS	Configuration server (switch or end system)
	Tells LEC who his LES is
LES	Server (switch or end system)
	ARP like functions
BUS	Broadcast and unknown server (switch or end system)
	Supports broadcasts

Only one LES / BUS per ELAN ==> put it in the switch

Operation

LECS connect phase

Get own ATM address and MAC address (part of ATM address)
Find LECS address
Use ilmi (to switch)
Use well known address
47:00:79:00:00:00:00:00:00:00:00:00:00:00:A0:3E:00:00:01:00
Use well know PVC (0, 17)
Call the LECS and establish the *configuration direct* VCC

Configuration phase

LEC –––– CFDVCC –––> LECS

ATM address

MAC address

LECS –––– CFDVCC –––> LEC

LES ATM addresss

ELAN type

MTU

Join Phase

Establish *control direct* VCC to LES

LEC ––– CDVCC –––> LES

LE_JOIN_REQUEST

ATM address

MAC address

LES ––– CDVCC –––> LEC

LE_JOIN_REPLY

(and add LEC to *control distribute* VCC)

BUS connect phase

LEC ––– CDVCC ––> LES

whois ff:ff:ff:ff:ff:ff

LES ––– CDVCC ––> LEC

ff:ff:ff:ff:ff:ff is ATM addresss

Establish *multicast send* VCC to BUS

BUS adds LEC to its multicast forward VCC

Data connects

LEC ––– CDVCC ––> LES

whois 00:12:34:65 (mac address)

LES ––– CDVCC ––> LEC

00:12:34:65 is ATM addresss

Establish *data direct* VCC to ATM address

How is mac address learned??

Regular ARP

How does regular ARP work

Just like regular ARP –– via the BUS

Here is a log of the signalling exchange required in creating an atm connection:

```
16 (909619989.652531) FROM KERNEL:
  as_connect (vcc 0xc0cbc600, listen_vcc 0x0)
  reply 0, aal 5
  pvc 0.0.0
  local atm−atm
  qos ubr,aal5:max_pcr=120,max_sdu=1516
  svc 47.0203.04050607080900000101.0020359972B2.03
  sap blli:l3=tr9577,ipi=snap,oui=0x00A03E,pid=0x0001

17 (909619989.653161) TO NETWORK:
  _pdsc = 9 "Q.2931 user−network call/connection control message"
  _cr_len = 3
  call_ref = 1 (0x1)
  msg_type = 0x05 "SETUP"
  _ext = 1
  _flag = 0 "instruction field not significant"
  _action_ind = 0 "clear call"
  msg_len = 104 (0x68)
    _ie_id = 0x58 "ATM adaption layer parameters"
      _ext = 1
      _cs = 0 "ITU−T standardized"
      _flag = 0 "instruction field not significant"
      _action_ind = 0 "clear call"
      _ie_len = 11 (0xb)
        aal_type = 5
          _id = 0x8c "Forward maximum CPCS−SDU size"
            fw_max_sdu = 1516 (0x5ec)
          _id = 0x81 "Backward maximum CPCS−SDU size"
            bw_max_sdu = 1516 (0x5ec)
          _id = 0x83 "AAL mode (UNI 3.0 only)"
            aal_mode = 1 (0x1)
          _id = 0x84 "SSCS type"
            sscs_type = 0 (0x0)
    _ie_id = 0x59 "ATM traffic descriptor"
      _ext = 1
      _cs = 0 "ITU−T standardized"
      _flag = 0 "instruction field not significant"
      _action_ind = 0 "clear call"
      _ie_len = 9 (0x9)
        _id = 0x84 "Forward peak cell rate (CLP=0+1)"
          fw_pcr_01 = 120 (0x78)
        _id = 0x85 "Backward peak cell rate (CLP=0+1)"
          bw_pcr_01 = 120 (0x78)
  _id = 0xbe "Best effort indicator"
_ie_id = 0x5e "Broadband bearer capability"
 _ext = 1
 _cs = 0 "ITU−T standardized"
 _flag = 0 "instruction field not significant"
 _action_ind = 0 "clear call"
 _ie_len = 3 (0x3)
  bearer_class = 16 "BCOB−X"
  _ext = 0
   _ext = 1
   trans_cap = 0x00 "Non−real time VBR (reception only)"
  _ext = 1
  susc_clip = 0 "not susceptible to clipping"
  upcc = 0 "point−to−point"
_ie_id = 0x5f "Broadband low−layer information"
 _ext = 1
 _cs = 0 "ITU−T standardized"
 _flag = 0 "instruction field not significant"
 _action_ind = 0 "clear call"
 _ie_len = 9 (0x9)
  _lid = 3
   uil3_proto = 0x0b "ISO/IEC TR 9577"
    _ext = 0
     _ext = 0
     ipi_high = 0x40
      _ext = 1
      _ipi_low = 0
       _ext = 1
       _snap_id = 0
       oui = 41022 (0xa03e)
       pid = 1 (0x1)
_ie_id = 0x70 "Called party number"
 _ext = 1
 _cs = 0 "ITU−T standardized"
 _flag = 0 "instruction field not significant"
 _action_ind = 0 "clear call"
 _ie_len = 21 (0x15)
  _ext = 1
  _plan = 2 "ATM endsystem address"
   _type = 0 "unknown"
   cdpn_esa = 47 2 3 4 5 6 7 8 9 0 0 1 1 0 20 35 99 72 b2 3
_ie_id = 0x6c "Calling party number"
 _ext = 1
 _cs = 0 "ITU−T standardized"
 _flag = 0 "instruction field not significant"
 _action_ind = 0 "clear call"
```

```
      _ie_len = 21 (0x15)
        cgpn_plan = 2 "ATM endsystem address"
        cgpn_type = 0 "unknown"
        _ext = 1
        cgpn = 47 2 3 4 5 6 7 8 9 0 0 1 4 0 4 ac 6c 28 f1 0
    _ie_id = 0x5c "Quality of service parameter"
      _ext = 1
      _cs = 3 "Standard defined for the network"
      _flag = 0 "instruction field not significant"
      _action_ind = 0 "clear call"
      _ie_len = 2 (0x2)
        qos_fw = 0
        qos_bw = 0

18 (909619989.731295) FROM NETWORK:
  _pdsc = 9 "Q.2931 user−network call/connection control message"
  _cr_len = 3
  call_ref = 8388609 (0x800001)
  msg_type = 0x02 "CALL PROCEEDING"
  _ext = 1
  _flag = 0 "instruction field not significant"
  _action_ind = 0 "clear call"
  msg_len = 9 (0x9)
    _ie_id = 0x5a "Connection identifier"
      _ext = 1
      _cs = 0 "ITU−T standardized"
      _flag = 0 "instruction field not significant"
      _action_ind = 0 "clear call"
      _ie_len = 5 (0x5)
        _ext = 1
        _vp_ass = 1 "explicit indication of VPCI"
        _pref_exc = 0 "exclusive VPCI; exclusive VCI"
        vpi = 0 (0x0)
        vci = 48 (0x30)

19 (909619989.773095) FROM NETWORK:
  _pdsc = 9 "Q.2931 user−network call/connection control message"
  _cr_len = 3
  call_ref = 8388609 (0x800001)
  msg_type = 0x07 "CONNECT"
  _ext = 1
  _flag = 0 "instruction field not significant"
  _action_ind = 0 "clear call"
  msg_len = 9 (0x9)
    _ie_id = 0x5a "Connection identifier"
      _ext = 1
```

```
136      _cs = 0 "ITU−T standardized"
137      _flag = 0 "instruction field not significant"
138      _action_ind = 0 "clear call"
139      _ie_len = 5 (0x5)
140        _ext = 1
141        _vp_ass = 1 "explicit indication of VPCI"
142        _pref_exc = 0 "exclusive VPCI; exclusive VCI"
143        vpi = 0 (0x0)
144        vci = 48 (0x30)

145 20 (909619989.773362) TO NETWORK:
146   _pdsc = 9 "Q.2931 user−network call/connection control message"
147   _cr_len = 3
148   call_ref = 1 (0x1)
149   msg_type = 0x0f "CONNECT ACKNOWLEDGE"
150   _ext = 1
151   _flag = 0 "instruction field not significant"
152   _action_ind = 0 "clear call"
153   msg_len = 0 (0x0)
154 21 (909619989.773731) TO KERNEL:
155   as_okay (vcc 0xc0cbc600, listen_vcc 0x0)
156   reply 0, aal 5
157   pvc 0.0.48
158   local atm−atm
159   qos ubr,aal5:max_pcr=120,max_sdu=1516
160   svc <invalid>
161   sap blli:l3=tr9577,ipi=snap,oui=0x00A03E,pid=0x0001

497 ilmid:IO: SNMP message received:
498 { −− SEQUENCE −−
499     version 0,
500     community ’494c4d49’H  −− "ILMI" −−,
501     data set−request { −− SEQUENCE −−
502           request−id 10,
503           error−status 0,
504           error−index 0,
505           variable−bindings { −− SEQUENCE OF −−
506              { −− SEQUENCE −−
507              name {1 3 6 1 4 1 353 2 7 1 1 3 0 13 71 2 3 4 5 6 7 8 9 0 0 1 4},
508                 value simple number 1
509              }
510           }
511       }
512 }
```

```
530 ilmid:ILMI: switch registered a network prefix
531 ilmid:ILMI: entering state ADDRESS
532 ilmid:ILMI: setting the atm address on the switch
533 ilmid:IO: SNMP message sent:
534 { –– SEQUENCE ––
535    version 0,
536    community ’494c4d49’H  –– "ILMI" ––,
537    data set–request { –– SEQUENCE ––
538          request–id 4,
539          error–status 0,
540          error–index 0,
541          variable–bindings { –– SEQUENCE OF ––
542             { –– SEQUENCE ––
543                name {1 3 6 1 4 1 353 2 6 1 1 3 0 20 71 2 3 4 5 6 7 8 9 0 0 1 4 0 4 172
                                                     108 40 241 0},
544
545                value simple number 1
546             }
547          }
548       }

566 ilmid:ILMI: received get response
567 ilmid:ILMI: ATM address registered
568 ilmid:IO: Primary ATM Address 47.0203.040506070809000000104.0004AC6C28F1.00
      added l
569 ilmid:ILMI: entering state POLL
570 ilmid:ILMI: sending get–next request
571 ilmid:IO: SNMP message sent:
```

```
 13 LAN Emulation Frame Xmitted: (143 bytes)
 14
 15       FF00 – Control Frame Marker
 16         01 – Protocol
 17         01 – Version
 18       0002 – OpCode (LE_JOIN_REQUEST)
 19       0000 – Status (Success)
 20   00640305 – Transaction ID
 21       0000 – Requestor LEC_ID
 22       0000 – Flags
 23   Source LAN Destination : MAC : 00–04–ac–6c–28–f1
 24   Source ATM : pre:47020304050607080900000104 esi:00–04–ac–6c–28–f1 sel:00
 25 CM     EVNT ’conn.c’: Incoming call
 26 LECCTL EVNT ’lec_ctrl.c’: Accepting Control Distribute VCC.
 28 LAN Emulation Frame Received: (108 bytes)
 29
 30       FF00 – Control Frame Marker
 31         01 – Protocol
 32         01 – Version
 33       0102 – OpCode (LE_JOIN_RESPONSE)
 34       0000 – Status (Success)
 35   00640305 – Transaction ID
 36       000e – Requestor LEC_ID
 37       0000 – Flags
 38   Source LAN Destination : MAC : 00–04–ac–6c–28–f1
 39   Source ATM : pre:47020304050607080900000104 esi:00–04–ac–6c–28–f1 sel:00
 40 LECCTL TRAN ’lec_ctrl.c’: (S_JOIN_WAIT, E_RCV_JOIN_RSP) –>
S_BUS_ARP_WAIT

 42 LAN Emulation Frame Xmitted: (108 bytes)
 43
 44       FF00 – Control Frame Marker
 45         01 – Protocol
 46         01 – Version
 47       0006 – OpCode (LE_ARP_REQUEST)
 48       0000 – Status (Success)
 49   00640306 – Transaction ID
 50       000e – Requestor LEC_ID
 51       0000 – Flags
 52   Source LAN Destination : <Not Present>
 53   Target LAN Destination : MAC : ff–ff–ff–ff–ff–ff
 54   Source ATM : pre:47020304050607080900000104 esi:00–04–ac–6c–28–f1 sel:00
 55   Target ATM : pre:00000000000000000000000000 esi:00–00–00–00–00–00 sel:00
```

```
57 LAN Emulation Frame Received: (108 bytes)
58
59       FF00 – Control Frame Marker
60         01 – Protocol
61         01 – Version
62       0106 – OpCode (LE_ARP_RESPONSE)
63       0000 – Status (Success)
64   00640306 – Transaction ID
65       000e – Requestor LEC_ID
66       0000 – Flags
67   Source LAN Destination : <Not Present>
68   Target LAN Destination : MAC : ff–ff–ff–ff–ff–ff
69   Source ATM : pre:47020304050607080900000104 esi:00–04–ac–6c–28–f1 sel:00
70   Target ATM : pre:47020304050607080900000101 esi:00–20–35–99–72–b2 sel:03
71 LECCTL TRAN ’lec_ctrl.c’: (S_BUS_ARP_WAIT, E_RCV_BUS_ARP_RSP) –>
     S_BUS_SVC_WAIT
72 CM     EVNT ’conn.c’: Outgoing call setup
73 CM     EVNT ’conn.c’: Call to 47 02 03 04 05 06 07 08 09 00 00 01 01 00 20 35 99 7
74 LECCTL EVNT ’lec_ctrl.c’: Multicast Send VCC Ready
75 LECCTL TRAN ’lec_ctrl.c’: (S_BUS_SVC_WAIT, E_BUS_SVC_READY) –>
     S_OPERATIONAL
76 CM     EVNT ’conn.c’: Incoming call
77 LECCTL EVNT ’lec_ctrl.c’: Accepting Multicast Distribute VCC.
78 CM     EVNT ’conn.c’: Call from 47 02 03 04 05 06 07 08 09 00 00 01 01 00 20 35 99

80 LAN Emulation Frame Xmitted: (108 bytes)
81
82       FF00 – Control Frame Marker
83         01 – Protocol
84         01 – Version
85       0006 – OpCode (LE_ARP_REQUEST)
86       0000 – Status (Success)
87   00640307 – Transaction ID
88       000e – Requestor LEC_ID
89       0000 – Flags
90   Source LAN Destination : MAC : 00–04–ac–6c–28–f1
91   Target LAN Destination : MAC : 00–e0–a3–55–58–00
92   Source ATM : pre:47020304050607080900000104 esi:00–04–ac–6c–28–f1 sel:00
93   Target ATM : pre:00000000000000000000000000 esi:00–00–00–00–00–00 sel:00
```