

# ITCH Multicast Offering



**January 12, 2011**

Version 1.5

**NASDAQ OMX<sup>SM</sup>**

# INET Nordics ITCH flow based on IP Multicasting

**Production** addresses & ports: **page 5 & 9** (and in a separate document for "Premium Co-location Customers").

**Test** addresses & ports: **page 13**

Apart from INET Nordics providing the ITCH flow over TCP connections, ITCH is also provided as an IP Multicast flow (and with UDP). For a Participant with many ITCH clients, the network load (i.e. bandwidth demand) is less if IP Multicasting is used, as packet being sent out once, is received by all clients. The following pages explain ITCH based on IP Multicasting.

General principles:

- INET Nordics provides Production ITCH flow over UDP/IP Multicast from two sites simultaneously. For the TEST Multicast flow, there is also two flows, but both originates from same site.
- The flow from the Primary site shall normally only be used, and Secondary flow used if Primary fails. It is though possible to use the flow from the Secondary site at any time. Co-location customer can however only use the Site A flow.
- A client application may also receive both flows simultaneously, and discard duplicates. The side effect is extra cpu load, but enhances the reliability.
- A client application joins a Multicast group by means of IGMP (Internet Group Management Protocol). Nowadays all TCP/IP stack implementations includes IGMP. RFC 1112 ("Host Extension for IP Multicasting") describes IGMPv1. IGMPv2 and IGMPv3 are described in later RFCs. RFC 3376 is the latest.
- PIM-DM (Dense Mode, and hence no RP) is used. An Extranet provider may change to Sparse mode (check with the Extranet provider what applies).
- As opposed to TCP, UDP on the receive side cannot detect lost packets and request retransmission (where TCP uses the SACK procedures), and UDP on the sending side has no timer for retransmission (where TCP runs a timer waiting for ACK). Therefore the application protocol needs to have a mechanism (such as sequence number) for detecting lost messages and taking recovery action.
- MoldUDP is the INET Nordics application protocol for handling sequence numbering. "UDP" in the name denotes that it uses UDP as the transport protocol. One or several ITCH messages is contained in a MoldUDP packet.

# A general overview to explain the Market Data protocols from a layered perspective.

<p><b>Market Data Products</b></p>	<p><b>Market feeds</b> (provided through different TCP ports):</p> <ul style="list-style-type: none"> <li>• All INET Market Data</li> <li>• Market Data Equities and Related</li> <li>• Market Data Warrants</li> <li>• Other Market Data feeds</li> </ul>	<p><b>Market feed:</b></p> <ul style="list-style-type: none"> <li>• All INET Market Data</li> </ul>	<p><b>Market feeds</b> (shaped for each account):</p> <ul style="list-style-type: none"> <li>• Market Data Equities Limited</li> <li>• Market Data Equities Full</li> <li>• Market Data All Market</li> <li>• Market Data Fixed Income</li> <li>• Market Data Derivatives</li> <li>• Others</li> </ul>	
<p><b>5. Session and upper layers</b> DHCP · DNS · FTP · Gopher · HTTP · IMAP4 · IRC · NNTP · XMPP · POP3 · SIP · SMTP · SNMP · SSH · TELNET · RPC · RTCP · RTSP · TLS · SDP · SOAP · GTP · STUN · NTP · RIP · ...</p>	<p><b>ITCH</b></p>	<p><b>ITCH</b></p>	<p><b>TIP</b></p>	
<p><b>4. Transport layer</b> TCP · UDP · DCCP · SCTP · RTP · RSVP · IGMP · ICMP · ICMPv6 · PPTP · ...</p>	<p><b>SoupTCP</b></p>	<p><b>MoldUDP</b></p>	<p><b>XMP</b></p>	
<p><b>3. Network/Internet layer</b> IP (IPv4 · IPv6) · OSPF · IS-IS · BGP · IPsec · ARP · RARP · ...</p>	<p>IP</p>	<p>IP</p>	<p>IP</p>	
<p><b>2. Data link layer</b> 802.11 · Wi-Fi · WiMAX · ATM · DTM · Token Ring · Ethernet · FDDI · Frame Relay · GPRS · EVDO · HSPA · HDLC · PPP · L2TP · ISDN · ...</p>				
<p><b>1. Physical layer</b> Ethernet physical layer · Modems · PLC · SONET/SDH · G.709 · OFDM · Optical Fiber · Coaxial Cable · Twisted Pair · ...</p>				

This column applies to the ITCH IP Multicast service

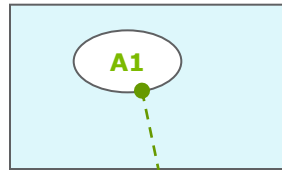
## An INET UDP flow Constraint

A Participant with a too low MTU size configured in the Participant's own network, cannot use the INET UDP flows. This has to do with the following facts:

- When multiple messages are available for dissemination in the ITCH Multicast server, they are batched to be encapsulated in the same MoldUDP Downstream packet (which hence contains multiple message blocks; number as indicated in the Message Count field).
- The maximum size of a MoldUDP Downstream packet is 1440 bytes. However, it may only contain complete messages (i.e. if a message block cannot fit in the MoldUDP packet, it is put in the next MoldUDP packet).
- A MoldUDP Downstream packet will be subject to the following headers added by UDP and IP: 8 + 20, which in turn may give the max size of  $1440 + 8 + 20 = 1468$  bytes.
- From the above, the following can be stated: if any hop along the route provides a MTU size lower than 1468 bytes, the MoldUDP packet will not arrive to its destination.

Not to get confused: low MTU sizes is not a problem for TCP traffic, as the "path MTU discovery mechanism" normally is implemented in the Participant's host, and TCP adopts its MSS based on the lowest MTU size discovered.

**ITCH server, Primary**

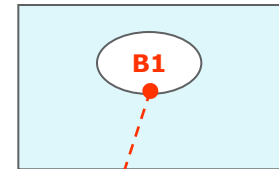


ITCH application sending messages according to the MoldUDP protocol. Sent out as UDP/IP Multicast packets

IP address for ITCH Multicast host (source IP address in the Multicast packets): **194.132.163.65**

NASDAQ OMX **Site A**  
Extranet subnet for IP Multicast service: **194.132.163.64/27**

**ITCH server, Secondary**



ITCH application as on Site A.

NASDAQ OMX **Site B**  
Extranet subnet for IP Multicast service: **192.165.254.32/27**

IP address for ITCH Multicast host (source IP address in the Multicast packets): **192.165.254.33**

**Multicast group A1:**  
Destination UDP port: **31041**  
Destination IP address (i.e. Multicast address): **233.74.125.41**

**Verizon's VFN or other Extranet provider's network**

An access link may only carry either the A or the B Production Multicast flow

**Multicast group B1:**  
Destination UDP port: **31141**  
Destination IP address (i.e. Multicast address): **233.74.125.141**

The router module in the respective CPE handles only the A or B Multicast group. As depicted, CPE1 is multicast router for A1 only, and CPE2 is multicast router for B1 only. If any CPE fails, only one of the Multicast groups is available.



Participant hosts acting as INET Nordics clients

Participant's clients receiving ITCH Multicast flows:

- Client "a": has joined 233.74.125.41
- Client "ab": has joined 233.74.125.41 and 233.74.125.141
- Client "b": has joined 233.74.125.141

## IP Multicast client actions (based on that UDP is used as Transport Protocol)

### ESTABLISHMENT PHASE

A client application for receiving the UDP/IP Multicast flow shall establish a connection for the use of UDP connectionless service, in combination with joining an IP Multicast group. This is normally done through the following steps:

- **Create a socket** by requesting UDP datagram type of socket (SOCK\_DGRAM)
- **Bind** the socket, whereby the socket is associated with the client's local address. The local port means here the UDP port that INET Nordics sends as destination port number. The local interface for receiving the IP Multicast traffic can be set to INADDR\_ANY, which assumes that the traffic is received over the default multicast i/f (behavior may vary with o/s).
- **Join the multicast group** through the socket option IP\_ADD\_MEMBERSHIP. The IP address is the Multicast group address (in space 233.74.125.n where "n" depends on which group to join). Manuals may refer to this address parameter as *imr\_multiaddr*. As opposed to the address parameter *imr\_interface* which is set to the client's local IP address (associated with the LAN i/f over which the IP Multicast traffic shall be received). It can be set to INADDR\_ANY which gives joining over the default multicast i/f; but therefore it is recommended to specify the IP address to make sure the correct i/f is used.

"Default multicast i/f" is the local interface where the multicast route (e.g. 224.0.0.0, mask 240.0.0.0) is set. Check with *netstat -rn*.

The IP Multicast address is a class-D address. INET Nordics uses globally unique addresses as defined in the GLOP addressing RFC (RFC2770).

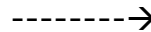
Joining a multicast group means that the client sends data to the CPE (router) according to IGMP (Internet Group Management Protocol), layered above IP like (ICMP) and with IP protocol number = 2. Thus, membership management between host and router is carried out by IGMP as on next page.

If joining fails, the client should try to join the other multicast group (i.e the one originating from the other INET Nordics site).

Leaving a multicast group is done by socket option IP\_DROP\_MEMBERSHIP (with parameters matching those set in IP\_ADD\_MEMBERSHIP).

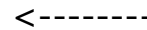
## IGMP principles, host - multicast router (*multicast router* means here the CPE router enabled to handle Multicast groups)

**Host:** sends IGMP report, action to join Multicast group  
(identified with the IP Multicast group address)

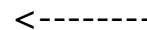


**Router:** sends no response according to IGMP.

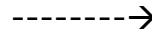
When packet has arrived from the source (i.e. from NASDAQ OMX), the router just sends it out on the local LAN. All clients having joined will receive it. Thus, it is sent out once on the LAN, regardless of the number of hosts having joined.



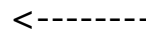
**Router:** may send IGMP query messages at regular intervals. To check if the host is still joined to the group).



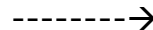
**Host:** sends IGMP response if still joined to the group



**Router,** now up and running again: starts with sending IGMP query.



**Host:** sends IGMP response as it is still joined to the group



No response if the host has made failover by joining the other group. If so, it has been based on application level decision (i.e message time out condition)

Comment 1: the above explains why the host (not knowing about the router being restarted) does not need to take any action, such as leaving and rejoining the group.

Comment 2: this page aims to explain IGMP traffic; if however the router operates in Dense mode, it just "pushes" out packets on the LAN regardless of the state of the host.

## Data Receiving Phase

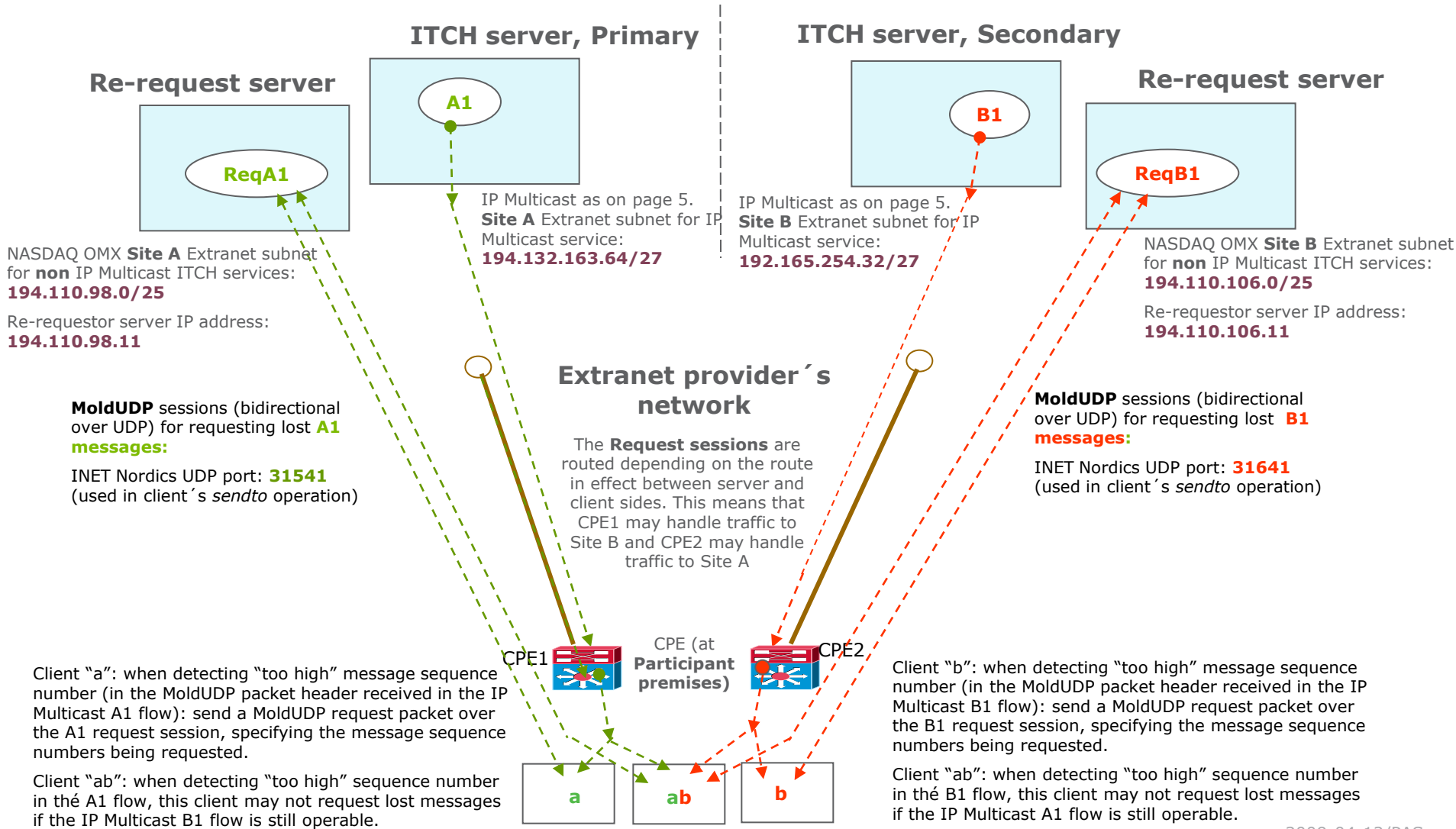
INET Nordics ITCH server puts one or several ITCH messages into a "MoldUDP packet". A MoldUDP packet is in turn delivered to the UDP layer which creates an UDP datagram. And in the IP layer, it is put into a IP datagram, where the destination IP address field is set to the IP Multicast group address.

At the receiving side (i.e. the ITCH client side), the application gets a MoldUDP packet from the UDP layer. As opposed to TCP, being so-called byte stream oriented, UDP delivers a service data unit to the application which is the same data unit as sent by the application source. The service data unit is thus the MoldUDP packet.

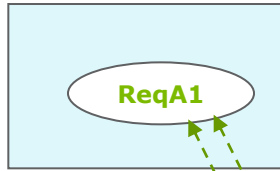
UDP has checksum and can thus prevent a corrupted datagram from being delivered to the application. As UDP (like IP) has no mechanisms for acknowledgement and flow control, a discarded packet cannot be resent and a "slow receiver" cannot pace the receive rate (whereby peaks may result in dropped packets, which also is the result when transmission disturbance occurs). It is the client application that needs to take action when there is a gap in the application sequence number. In this case according to the MoldUDP protocol. A "too high" sequence number means lost message(s), and recovery needs to be performed over a separate session; a request session (explained on next page). Such an event gives extra load to the client application as receiving of new ITCH messages take place at the same time as requested lost messages are received (on the request session). The request session is also UDP based (and not TCP), meaning that network/receive buffer problem can also cause the recovery data to get lost. Instead of repeating (after a period of time) the request for lost messages to the same site, it is better to repeat the request to the other site.

Please observe the meaning of *maximum payload size* in the "MoldUDP protocol specification". This size corresponds to the aggregated messages up to the limit for what can fit into a MoldUDP Downstream packet. The protocol spec. says: ***If the total size of the requested messages exceeds the maximum payload size of the server, only the number of messages that completely fit will be returned.*** Hence, the client only receives one MoldUDP Downstream packet (with multiple ITCH messages) after a re-request, and to get the next chunk of lost ITCH messages a new re-request (with a new seq. number) is needed, etc. This in turn means that one re-request sent to the server and one MoldUDP Downstream packet received from the server goes hand-in-hand. The method prevents a high re-request load from consuming much bandwidth. Thus, it does not give a load affecting the other Unicast traffic even if a very large seq. number gap needs to be recovered.

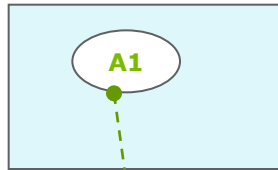
Failover to the other IP Multicast group is explained in subsequent pages (Failover case I and II in the header). If the client application only receives from one Multicast group at a time, the failover procedure should include leaving the current Multicast group.



**Re-request server**

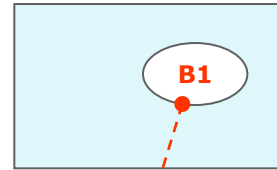


**ITCH server, Primary**



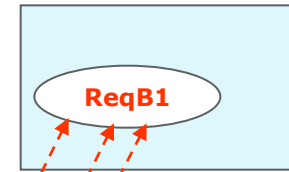
IP Multicast as on previous page. **Site A** Extranet subnet for IP Multicast service: **194.132.163.64/27**

**ITCH server, Secondary**



IP Multicast as on previous page. **Site B** Extranet subnet for IP Multicast service: **192.165.254.32/27**

**Re-request server**



NASDAQ OMX **Site B** Extranet subnet for **non** IP Multicast ITCH services: **194.110.106.0/25**

Re-requestor server IP address: **194.110.106.11**

NASDAQ OMX **Site A** Extranet subnet for **non** IP Multicast ITCH services: **194.110.98.0/25**

Re-requestor server IP address: **194.110.98.11**

**Multicast group B1:**  
Destination UDP port: **31141**  
Destination IP address (i.e. Multicast address): **233.74.125.141**

**MoldUDP** sessions (bidirectional over UDP) for requesting lost **A1** messages:

INET Nordics UDP port: **31541**  
(used in client's *sendto* operation)

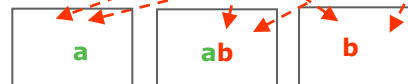
**MoldUDP** sessions (bidirectional over UDP) for requesting lost **B1** messages:

INET Nordics UDP port: **31641**  
(used in client's *sendto* operation)

**Extranet provider's network**



CPE (at Participant premises)

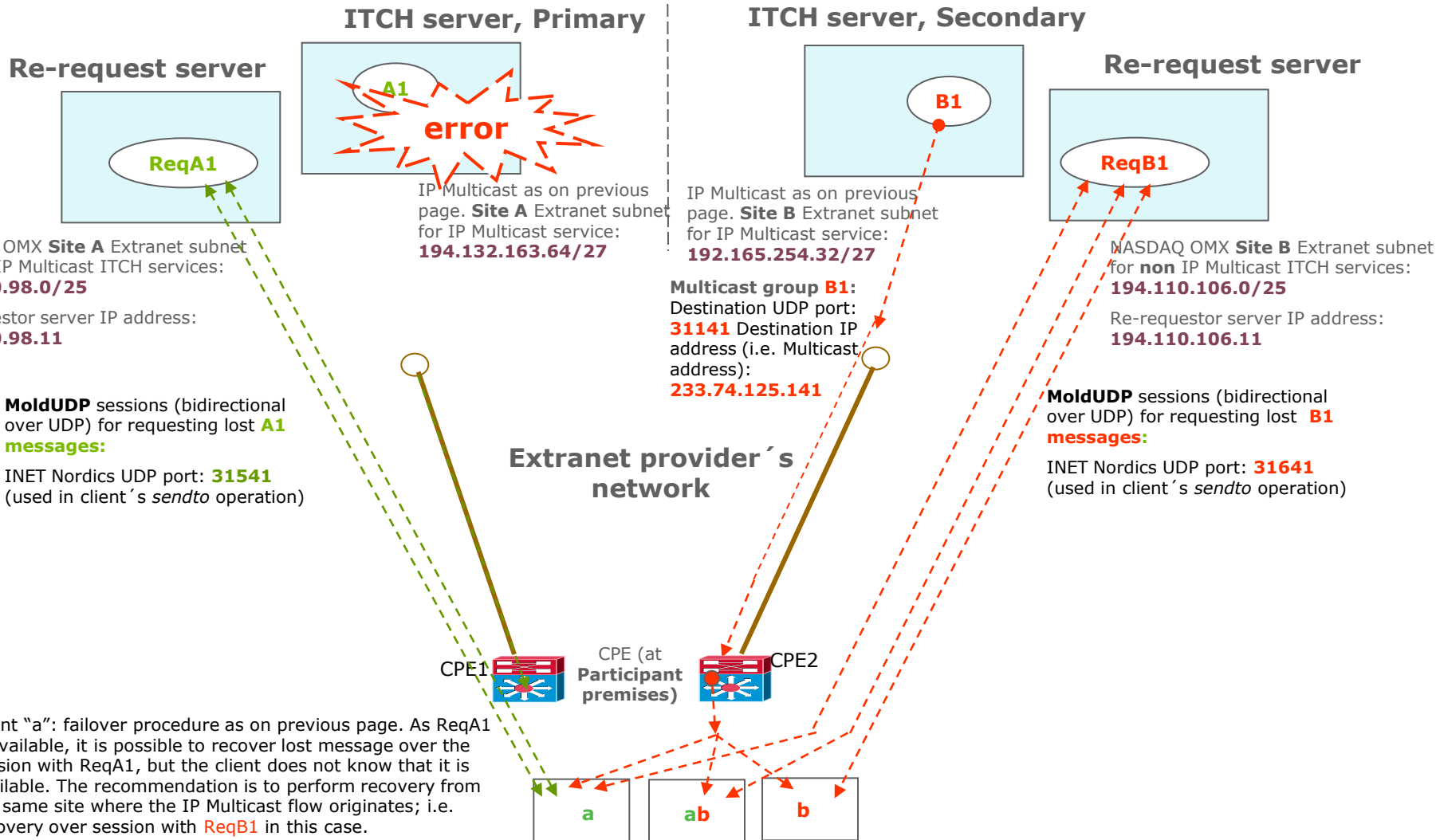


Client "a": after X number of seconds with no data received from Multicast group A1, failover is made to B1, (i.e. IP Multicast group B1 is joined). When receiving messages again, a seq. number gap is most likely experienced. Thus, recovery of lost messages is made (ref. the previous page), but to **ReqB1**.

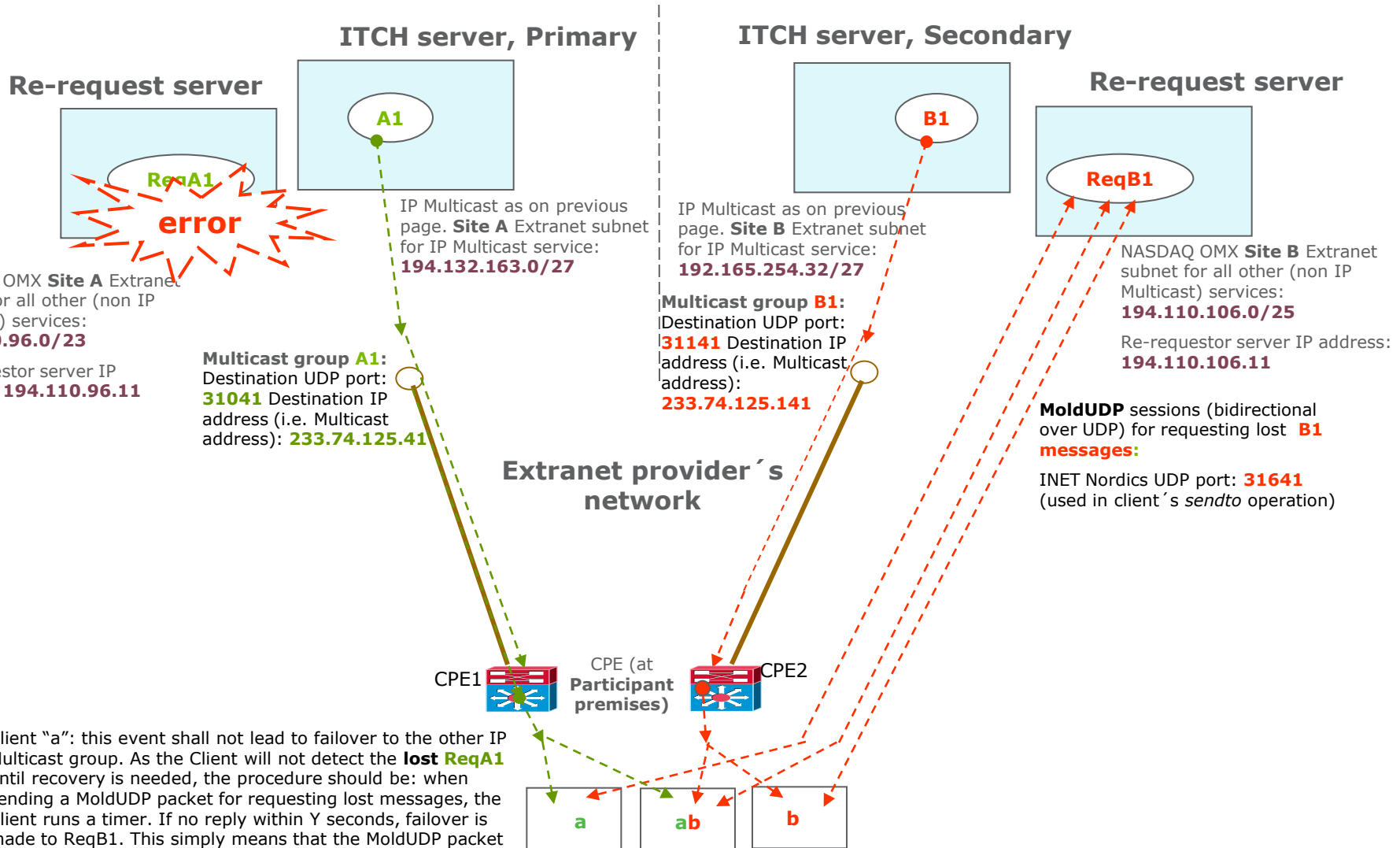
Client "ab": is already joined to B1. And stays joined to A1.

The MoldUDP session with ReqB1 for Client "a" (and ReqA1 for Client "b") is not depicted on the previous page. There are two choices: to initiate both the Request sessions at start-up time, or the second one only in case of failover. The first choice is recommended; i.e. a sessions to both ReqA1 and to ReqB1 is setup at start-up time.

ITCH server for A1 down, but Re-request server for A1 is available



Only Re-requestor server down



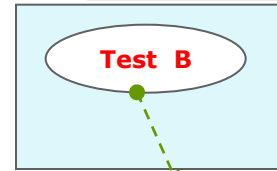
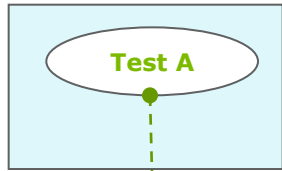
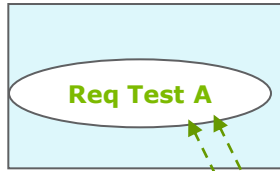
Client "a": this event shall not lead to failover to the other IP Multicast group. As the Client will not detect the **lost ReqA1** until recovery is needed, the procedure should be: when sending a MoldUDP packet for requesting lost messages, the Client runs a timer. If no reply within Y seconds, failover is made to ReqB1. This simply means that the MoldUDP packet is resent, but over the session to **ReqB1**.

This page shows INET Nordics **Test** Multicast flow; only available from Site A. But set up with two flows to provide failover testing. With IP addresses and UDP ports also for re-requests of lost Test messages

**ITCH Multicast servers – Test (at Site A only)**

N.B: the secondary flow is called **“Test B”** despite it sources from Site A.

**Re-request server - Test**



NASDAQ OMX **Site A** Extranet subnet for IP Multicast **Test** services: **194.130.55.112/28**

IP address for ITCH Multicast **Test A** host (source IP address in the Multicast packets): **194.130.55.113**

IP address for ITCH Multicast **Test B** host: **194.130.55.123**

NASDAQ OMX **Site A** Extranet subnet for **non** IP Multicast **Test** services: **194.110.101.0/24**

Re-requestor server IP address: **194.110.101.48**

**Multicast group Test A:**  
Destination UDP port: **31045**  
Destination IP address (i.e. Multicast address): **233.74.125.45**

**Multicast group Test B:**  
Destination UDP port: **31044**  
Destination IP address (i.e. Multicast address): **233.74.125.44**

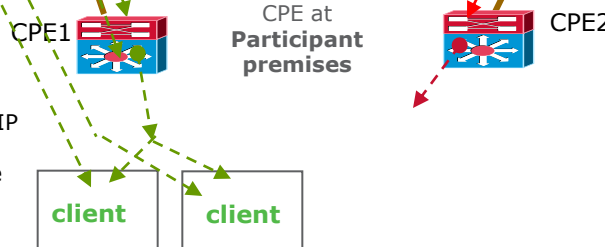
Secondary Re-requestor server IP address: **194.110.101.176** (subnet: **194.110.101.0/24**)

**MoldUDP** sessions for requesting lost **messages**. UDP port in the above host: **31544** (used in client's *sendto* operation)

**MoldUDP** sessions (bidirectional over UDP) for requesting lost **Test A messages**:  
INET Nordics UDP port: **31545** (used in client's *sendto* operation)

**Verizon's VFN or other Extranet provider's network**

Verizon provides the **Test B** IP Multicast flow via the Site B link (as here depicted). This is for making failover tests more production like. It is up to the respective Extranet provider to choose how this IP Multicast flow shall be distributed.



Clients: when detecting "too high" message sequence number (in the MoldUDP packet header received in the IP Multicast Test A flow): send a MoldUDP request packet over the Test A request session, specifying the message sequence numbers being requested.

## Multicast on the LAN (Ethernet) layer

(explains the relationship between the Layer 3 and Layer 2 multicast addresses)

---

All members having joined a Multicast group receive the same IP Multicast packet. But this is not done by using Ethernet broadcast addresses; it is done by using Ethernet multicast address. When the multicast router's LAN layer encapsulates the IP Multicast packet into a Ethernet frame, the destination MAC address field is hence set to the Ethernet multicast address. And this is the address that all members' LAN cards are set to accept incoming frames to.

By default, a host can receive LAN traffic to its own Ethernet MAC address (link station address) and to the Ethernet broadcast address. But the Ethernet multicast address(es) required must be set. This is not done by manual configuration, but automatically performed as the Layer 2 multicast address is derived from the Layer 3 multicast address. Or expressed as follows: the Ethernet multicast address is generated by basing it on a bit string from the IP Multicast address: the 23 lowest bits in the IP Multicast address is put into the 23 lowest bits of the 48 bits Ethernet address field.

## Discovery of UDP traffic problems

Like TCP , there is UDP statistics to analyze through the command ***netstat -s***.

UDP oriented events that may give discarded data (apart from discards based on faults found by the IP and LAN layers):

- Bad checksum
- Invalid header
- Bad data length
- Overflow (receive buffer)

Name of the statistic counters vary depending on the provider of the O/S (TCP/IP stack). Linux systems seem to only have a counter named "receive errors" for UDP problems.

N.B: the statistics is based on all traffic. So in this case, problem with other kind of UDP traffic will also give increased counters.

In some systems it is advised to increase the UDP receive buffer.

***netstat -s*** also shows IGMP statistics (if only watching this, use ***netstat -s -P igmp***)

Command ***netstat -g*** also gives information: what Multicast groups you have joined.

# Revision History

---

- V 1.1 Slide 4 added about UDP constraint
- V 1.2 Updates in technical descriptions including IP addresses added on some pages
- V 1.3 Site B address corrected
- V 1.4
  - a) Pages regarded as redundant have been deleted (and most pages thereafter renumbered)
  - b) Explanation that Dense mode (no RP) is used
  - c) IP Multicast for Test has been enhanced with a secondary flow
- V 1.5 Page 8 (Data Receiving Phase): to clarify that maximum payload size in the server gives the need of repeated re-requests.