ITCH Multicast for INET (Nordic) NLS feed

N.B: this document applies to the INET system (Equities market) NLS (Nordic Last Sale) feed; separate documents explain the INET “main” feed and ITCH Multicast for the Genium INET system.

This NLS related document contains same text as the document for the INET “main” feed, but with different short name, IP addresses and UDP ports.
INET Nordic ITCH flow based on IP Multicasting

- **Production** addresses & ports: page 6 & 10; and for Co-location customers: page 19 (1Gbps) and 20 (10Gbps)
- **Test** addresses & ports: page 16; and for Co-location customers: page 21
- To clarify the IP Multicast flows further distributed to NODE in UK and DE, NODE related pages are also found here below.

Apart from INET Nordic 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 a packet being sent out once could be received by multiple clients. The following pages explain ITCH based on IP Multicasting.

General principles:

- INET Nordic provides Production ITCH flow over UDP/IP Multicast from two sites simultaneously. For Colo customers only one site (Site A) applies, but with redundant Multicast flows. The same applies to the Test Multicast flow.
- The Primary flow shall normally only be used, and Secondary flow used if Primary fails. It is though possible to use the Secondary flow at any time, regardless of whether it is accessed at Site B or accessed as a redundant flow at Site A.
- 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.
- Nasdaq pushes out the Multicast flow as for PIM-DM (Dense Mode, and hence no RP applies). An Extranet provider may change to Sparse mode (check with the Extranet provider what applies). For those peering directly with Nasdaq SE Site A, option to use PIM-SM is offered. This in turn brings a RP address need, as explained on the next page.
- “Extranet provider” is used here to denote the external connectivity provider who receives the IP Multicast flows and further forwards out to its customers. As opposed to the kind of set up referred to as “Direct connect”, where the customer has its own L3 equipment directly connected to Nasdaq, but running over a telco (L2) service.
- As opposed to TCP, UDP on the receiving side has no procedure for lost packet detection and retransmission request (where TCP may use the SACK procedure), and neither has UDP on the sending side a 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.
- MoldUDP64 is the INET Nordic application protocol for handling sequence numbering. “UDP” in its name denotes that it uses UDP as the transport protocol. One or several ITCH messages are contained in a MoldUDP64 packet.
**INET Nordic ITCH Multicast based on PIM-SM (Sparse Mode)**

This only applies to SE site A. I.e. not to Site B and not to the NODE sites in UK and DE.

PIM-SM (Protocol Independent Multicast – Sparse Mode) is further explained in RFC 4601.

PIM-SM means that the peering party’s L3 switch/router sends a *PIM-SM join message* over the handoff to Nasdaq, where the message contains the applicable Rendezvous Point (RP) address as below.

**RPs at Site A (only):**

- For **1G Production, A flow**: 159.79.85.252/32
- For **1G Production, C flow**: 159.79.85.253/32
- For **1G Test, A flow**: 159.79.85.254/32
- For **1G Test, B flow**: 159.79.85.255/32
- For **10G Production, A flow**: 159.79.85.252/32 (same as for 1G Prod above)
- For **10G Production, C flow**: 159.79.85.253/32 (same as for 1G Prod above)

Nasdaq will advertise the above host specific routes only in the case where PIM-SM will be used.
## Market Data Products

- **Market feeds (provided through different TCP ports):**
  - All INET Market Data
  - Market Data Equities and Related
  - Market Data Warrants
  - Other Market Data feeds: e.g. NLS (Last Sale)

- **Market feed:**
  - All INET Market Data
  - INET NLS (Last Sale)

- **Market feed:**
  - Two asset classes, each one providing its own Genium INET Market Data feed:
    - Derivatives
    - Commodities
  - And for the same two asset classes, each one having its AMD feed

- **Market feed (shaped for each account):**
  - MBL with top 5 levels. For all markets or for specific markets

- **Market feeds (shaped for each account):**
  - Market Data Equities Limited
  - Market Data Equities Full
  - Market Data All Market
  - Market Data Fixed Income
  - Market Data Derivatives
  - Others

## Session and upper layers

<table>
<thead>
<tr>
<th>ITCH (for INET)</th>
<th>ITCH (for INET)</th>
<th>ITCH (for Genium INET)</th>
<th>OMnet – Broadcast messages</th>
<th>TIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoupBinTCP</td>
<td>MoldUDP64</td>
<td>MoldUDP64</td>
<td>With a sublayer handling &quot;polling&quot; from client to get messages</td>
<td>SoupBinTCP</td>
</tr>
</tbody>
</table>

## Transport layer

<table>
<thead>
<tr>
<th>TCP</th>
<th>UDP</th>
<th>UDP</th>
<th>TCP</th>
<th>TCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoupBinTCP</td>
<td>MoldUDP64</td>
<td>MoldUDP64</td>
<td>TIP</td>
<td>SoupBinTCP</td>
</tr>
</tbody>
</table>

## Network/Internet layer

<table>
<thead>
<tr>
<th>IP (IPv4 · IPv6) · OSPF · IS-IS · BGP · IPsec · ARP · RARP · ...</th>
</tr>
</thead>
</table>

## Data link layer

<table>
<thead>
<tr>
<th>802.11 · Wi-Fi · WiMAX · ATM · DTM · Token Ring · Ethernet · FDDI · Frame Relay · GPRS · EVDO · HSPA · HDLC · PPP · L2TP · ISDN ·</th>
</tr>
</thead>
</table>

## Physical layer

<table>
<thead>
<tr>
<th>Ethernet physical layer · Modems · PLC · SONET/SDH · G.709 · OFDM</th>
</tr>
</thead>
</table>

The **ITCH** protocol for INET is also referred to as “Nordic Equity TotalView – ITCH”. Right column is the **INET** IP Multicast service.

This column applies to the **Genium INET** IP Multicast service.

“Broadcast” above is for defining message types, and does not mean “broadcast” addressing.
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 MoldUDP64 Downstream packet (which hence contains multiple message blocks; number as indicated in the Message Count field).

- The maximum size of a MoldUDP64 Downstream packet is 1472 bytes. However, it may only contain complete messages (i.e. if a message block cannot fit in the MoldUDP64 packet, it is put in the next MoldUDP64 packet).

- A MoldUDP64 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 1472 + 8 + 20 = 1500 bytes.

- From the above, the following can be stated: if any hop along the route provides a MTU size lower than 1500 bytes, a max sized MoldUDP64 packet will not arrive to its destination.

Just to compare with TCP: low MTU size may not be a problem in the case of TCP traffic, as the “path MTU discovery mechanism” normally is implemented in the hosts, and the sending TCP adopts its max data size based on the lower MTU size discovered. This adoption may however only work if the ICMP packet that reports the MTU problem reaches back to the sending host, and is not discarded by a firewall rule or access-list.
**INET Nordic ITCH NLS flow based on IP Multicast – UDP ports and IP addresses as for Production**

**SE - Site A**

**ITCH server, Primary**

ITCH application sending messages according to the MoldUDP64 protocol. Sent out as UDP/IP Multicast packets.

**Multicast group A-Ls:**
Destination UDP port: 31034
Destination IP address (i.e. Multicast address): 233.74.125.34

IP address for ITCH Multicast host (source IP address in the Multicast packets): 159.79.85.11

Nasdaq Site A Extranet subnet for IP Multicast service: 159.79.85.0/28

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

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

**SE - Site B**

**ITCH server, Secondary**

ITCH application as on Site A.

**Multicast group B-Ls:**
Destination UDP port: 31142
Destination IP address (i.e. Multicast address): 233.74.125.142

IP address for ITCH Multicast host (source IP address in the Multicast packets): 192.165.254.59

Nasdaq Site B Extranet subnet for IP Multicast service: 192.165.254.32/27

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

**Participant’s clients receiving ITCH Multicast flows:**

Client “a”: has joined 233.74.125.34
Client “ab”: has joined 233.74.125.34 and 233.74.125.142
Client “b”: has joined 233.74.125.142
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 Nordic 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_multiaaddr`. 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 Nordic uses globally unique addresses as defined in the GLOP addressing RFC (RFC2770).

For simplicity reasons, the pictures here only show the router denoted as CPE (deployed by the Extranet provider), but participants may have own routers as intermediate nodes. Joining a multicast group means that the client sends data to the 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 Nordic 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
(which means here the router enabled for the Multicast group and on the same LAN as the host)

**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), 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.

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

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

**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 and with static configuration to push out a Multicast flow, traffic for this flow will be sent out even if no IGMP join has been received by the router.
Data Receiving Phase

The INET Nordic ITCH server puts one or several ITCH messages into a “MoldUDP64 packet”. A MoldUDP64 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. Each site (and A and B) has its own ITCH server, but sends out the same ITCH messages. However, the number of MoldUDP64 packets are not the same. E.g. Site B generally puts more ITCH messages in a MoldUDP64 packet; and based on an intersite hop for the B flow; site A is a bit ahead when sending out the ITCH messages.

At the receiving side (i.e. the ITCH client side), the application gets a MoldUDP64 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 MoldUDP64 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 MoldUDP64 protocol. A “too high” sequence number means lost message(s), and recovery needs to be performed over a separate session; a re-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 re-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 “MoldUDP64 protocol specification”. This size corresponds to the aggregated messages up to the limit for what can fit into a MoldUDP64 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 MoldUDP64 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 MoldUDP64 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 Uncast 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.
**INET Nordic ITCH NLS Re-request flow**

UDP sessions (for requesting lost messages in the IP Multicast flow) – UDP ports and IP addresses as for Production

---

**SE - Site A**

**Re-request server**

**ITCH server, Primary**

**A-Ls**

IP Multicast as on page 6.

**Site A** Extranet subnet for IP Multicast service:

159.79.85.0/28

**Extranet provider’s network**

The Request sessions are routed depending on the route in effect between server and client sides. This means that CPE1 may handle traffic to Site B and CPE2 may handle traffic to Site A

Client “a”: when detecting “too high” message sequence number (in the MoldUDP64 packet header received in the IP Multicast A flow): send a MoldUDP64 request packet over the A request session, specifying the message sequence numbers being requested.

Client “ab”: when detecting “too high” sequence number in the A flow, this client may not request lost messages if the IP Multicast B flow is still operable.

---

**SE - Site B**

**Re-request server**

**ITCH server, Secondary**

**B-Ls**

IP Multicast as on page 6.

**Site B** Extranet subnet for IP Multicast service:

192.165.254.32/27

**MoldUDP64 sessions (bidirectional over UDP) for requesting lost B-Ls messages:**

INET Nordic UDP port: 31642 (used in client’s sendto operation)

---

**INET – ITCH**

---

**Nasdaq Site A** Extranet subnet for non IP Multicast services:

159.79.81.0/25

Re-requestor server IP address:

159.79.81.34

MoldUDP64 sessions (bidirectional over UDP) for requesting lost A-Ls messages:

INET Nordic UDP port: 31534

(used in client’s sendto operation)

---

**Nasdaq Site B** Extranet subnet for non IP Multicast services:

194.110.106.0/25

Re-requestor server IP address:

194.110.106.42

MoldUDP64 sessions (bidirectional over UDP) for requesting lost B-Ls messages:

INET Nordic UDP port: 31642 (used in client’s sendto operation)

---

Client “b”: when detecting “too high” message sequence number (in the MoldUDP64 packet header received in the IP Multicast B flow): send a MoldUDP64 request packet over the B request session, specifying the message sequence numbers being requested.

Client “ab”: when detecting “too high” sequence number in the B flow, this client may not request lost messages if the IP Multicast A flow is still operable.
INET Nordic ITCH NLS IP Multicast flow
Failover case I

SE - Site A

Re-request server

aunch server, Primary

IP Multicast as on previous page. Site A Extranet subnet for IP Multicast service: 159.79.85.0/28

A-Ls

ReqA-Ls

Extranet provider’s network

error

CPE1

CPE2

INET Nordic ITCH NLS IP Multicast flow
Failover case I

SE - Site B

SE - Site A

SE - Site B

Re-request server

ITCH server, Primary

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

Multicast group B-Ls:
Destination UDP port: 31142 Destination IP address (i.e. Multicast address): 233.74.125.142

B-Ls

ReqB-Ls

Re-request server

Nasdaq Site B Extranet subnet for non IP Multicast services: 194.110.106.0/25
Re-requestor server IP address: 194.110.106.42

Nasdaq Site A Extranet subnet for non IP Multicast services: 159.79.81.0/25
Re-requestor server IP address: 159.79.81.34

MoldUDP64 sessions (bidirectional over UDP) for requesting lost A-Ls messages:
INET Nordic UDP port: 31534 (used in client´s sendto operation)

MoldUDP64 sessions (bidirectional over UDP) for requesting lost B-Ls messages:
INET Nordic UDP port: 31642 (used in client´s sendto operation)

Client “a”: after X number of seconds with no data received from Multicast group A, failover is made to B-Ls, (i.e. IP Multicast group B-Ls 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 ReqB-Ls.

Client “ab”: is already joined to B-Ls. And stays joined to A.

The MoldUDP64 session with ReqB for Client “a” (and ReqA 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 ReqB is setup at start-up time.
Client “a”: failover procedure as on previous page. As ReqA is available, it is possible to recover lost message over the session with ReqA, but the client does not know that it is available. The recommendation is to perform recovery from the same site where the IP Multicast flow originates; i.e. recovery over session with ReqB-Ls in this case.
Client “a”: this event shall not lead to failover to the other IP Multicast group. As the Client will not detect the **lost ReqA-Ls** until recovery is needed, the procedure should be: when sending a MoldUDP64 packet for requesting lost messages, the Client runs a timer. If no reply within Y seconds, failover is made to ReqB. This simply means that the MoldUDP64 packet is resent, but over the session to **ReqB-Ls**.

**INET Nordic ITCH NLS IP Multicast flow**

**Failover case III**

Only Re-requestor server down

**SE - Site A**

**ITCH server, Primary**

- **A-Ls**
- IP Multicast as on previous page. **Site A** Extranet subnet for IP Multicast service: 159.79.85.0/28

**Multicast group A-Ls:**
- Destination UDP port: 31034
- Destination IP address (i.e. Multicast address): 233.74.125.142

**SE - Site B**

**ITCH server, Secondary**

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

**Multicast group B-Ls:**
- Destination UDP port: 31142
- Destination IP address (i.e. Multicast address): 233.74.125.142

**Re-request server**

**ReqA-Ls**

**error**

Nasdaq **Site A** Extranet subnet for non IP Multicast services 159.79.81.0/25

Re-requestor server IP address: 159.79.81.34

**Multicast group A-Ls:**
- Destination UDP port: 31034
- Destination IP address (i.e. Multicast address): 233.74.125.142

**Extranet provider´s network**

**INET Nordic UDP port: 31642**

(used in client´s sendto operation)

**CPE (at Participant premises)**

**CPE1**

**CPE2**

**a**  **ab**  **b**
**INET Nordic ITCH NLS for Production**

This page explains how the IP Multicast flows are distributed to the UK NODE access points (in London). Applicable only to “NODE customers”.

**UK Site - at Slough (LD4)**

**Re-request server**
- **ReqA-Ls**

**Multicast group A-Ls:**
- Destination UDP port: 31034
- Destination IP address (i.e. Multicast address): 233.74.125.34

**Multicast group B-Ls:**
- Destination UDP port: 31142
- Destination IP address (i.e. Multicast address): 233.74.125.142

**Nasdaq switches at the London (UK) sites.**
- The A Multicast group is accessed from the “A switch” at the UK sites, and the B Multicast group is accessed from the “B switch” at the UK sites.
- The flow is pushed out (as for Dense Mode). Customers need static routes for the Multicast source IP nets.

**Node network**

**SE - Site A**
- **Site A Extranet subnet for IP Multicast service:** 159.79.85.0/28
- ** ReqA-Ls**

**SE - Site B**
- **Site B Extranet subnet for IP Multicast service:** 192.165.254.32/27
- ** ReqB-Ls**

**Note 1:**
**MoldUDP64** sessions (bidirectional over UDP) for requesting lost A-Ls messages:
- INET Nordic UDP port: 31534 (used in client’s sendto operation)
- This unicast traffic is possible via all four Nasdaq switches (i.e. via both A and B at the two UK sites)

**Note 2:**
**MoldUDP64** sessions (bidirectional over UDP) for requesting lost B-Ls messages:
- INET Nordic UDP port: 31642 (used in client’s sendto operation)
- This unicast traffic is possible via all four Nasdaq switches (i.e. via both A and B at the two UK sites)
This page explains how the IP Multicast flows are distributed to the DE NODE access point (in Frankfurt FR2). Applicable only to “NODE customers”.

**Re-request server**

- **Site A** Extranet subnet for non IP Multicast services: 159.79.81.0/25
- Re-requestor server IP address: 159.79.81.34 *(Note 1)*

**Multicast group A:**
- Destination UDP port: 31034
- Destination IP address (i.e. Multicast address): 233.74.125.34

**-note:**

**Note 1:**
MoldUDP64 sessions (bidirectional over UDP) for requesting lost **A-Ls messages:**
INET Nordic UDP port: 31534 *(used in client’s sendto operation)*
This unicast traffic is possible via both the Nasdaq switches (i.e. via both A and B at the Frankfurt site)

**Re-request server**

- **Site B** Extranet subnet for non IP Multicast services: 194.110.106.0/25
- Re-requestor server IP address: 194.110.106.42 *(Note 2)*

**Multicast group B:**
- Destination UDP port: 31142
- Destination IP address (i.e. Multicast address): 233.74.125.142

**Note 2:**
MoldUDP64 sessions (bidirectional over UDP) for requesting lost **B-Ls messages:**
INET Nordic UDP port: 31642 *(used in client’s sendto operation)*
This unicast traffic is possible via both the Nasdaq switches (i.e. via both A and B at the Frankfurt site).
This page shows INET NLS Nordic 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)**

Re-request server - Test

**Test A-Ls**

IP address for ITCH Multicast Test A host (source IP address in the Multicast packets): **159.79.85.90**

Multicast group Test A-Ls: Destination UDP port: **31050**
Destination IP address (i.e. Multicast address): **233.74.125.50**

**Test B-Ls**

IP address for ITCH Multicast Test B host: **159.79.85.122**

Multicast group Test B-Ls: Destination UDP port: **31051**
Destination IP address (i.e. Multicast address): **233.74.125.51**

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

Clients: when detecting “too high” message sequence number (in the MoldUDP64 packet header received in the IP Multicast Test A flow): send a MoldUDP64 request packet over the Test A request session, specifying the message sequence numbers being requested.

**MoldUDP64** sessions (bidirectional over UDP) for requesting lost Test A-Ls messages:

INET Nordics UDP port: **31550**

(used in client’s sendto operation)

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

Clients: when detecting “too high” message sequence number (in the MoldUDP64 packet header received in the IP Multicast Test B flow): send a MoldUDP64 request packet over the Test B request session, specifying the message sequence numbers being requested.

**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.

**Requests** server IP address:

159.79.87.239 (subnet: 159.79.86.0/23)
INET Nordic NLS Test Multicast flow; only available from Site A.

Same as on previous page, but this page explains how the flows are distributed to the UK NODE access points (in London). Applicable only to “NODE customers”.

Note 1: MoldUDP64 sessions (bidirectional over UDP) for requesting lost Test A-Ls messages:
INET Nordic UDP port: 31550 (used in client’s sendto operation).
This unicast traffic is possible via all four Nasdaq switches (i.e. via both A and B at the two UK sites)

Note 2: MoldUDP64 sessions for requesting lost messages from the above Req Test host. UDP port: 31551 (used in client’s sendto operation).
This unicast traffic is possible via all four Nasdaq switches (i.e. via both A and B at the two UK sites)
**INET Nordic NLS Test Multicast flow; only available from Site A.**

Same as on previous page, but this page explains how the flows are distributed to the DE NODE access points (in Frankfurt FR2). Applicable only to "NODE customers".

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

Note 1: **MoldUDP64** sessions (bidirectional over UDP) for requesting lost Test A-Ls messages:
- INET Nordics UDP port: 31550 (used in client’s `sendto` operation).
- This unicast traffic is possible via **both** the Nasdaq switches (i.e. via both A and B) at the Frankfurt site.

Note 2: **MoldUDP64** sessions for requesting lost messages from the above Req Test host.
- UDP port: 31551 (used in client’s `sendto` operation).
- This unicast traffic is possible via **both** the NASDAQ OMX switches (i.e. via both A and B).
This page shows INET Nordic ITCH NLS for Production - Multicast flow applicable to Stockholm Co-location (Site A only) for 1G. (10G on next page) “A” flow is with same IP addresses (and ports) as earlier described. “C” is here provisioned as secondary for 1G Co-location customers.

**ETCH server, Primary**

**A-Ls**

IP address for ITCH Multicast Prod A host (source IP address in the Multicast packets): **159.79.85.11**

Multicast group Prod A-Ls:
- Destination UDP port: **31034**
- Destination IP address (i.e. Multicast address): **233.74.125.34**

**ITCH server, Secondary**

**C-Ls**

IP address for ITCH Multicast Prod C host: **159.79.85.27**

Multicast group Prod C-Ls:
- Destination UDP port: **31035**
- Destination IP address (i.e. Multicast address): **233.74.125.35**

**MoldUDP64 sessions** (bidirectional over UDP) for requesting lost **Prod A-Ls** messages:

- Genium INET Nordics UDP port: **31534** (used in client’s sendto operation)

**Nasdaq Site A network, with topology for 1G Colo**

**Re-request server**

**Req A-Ls**

Nasdaq Site A Extranet subnet for non IP Multicast Prod services: **159.79.81.0/25**

Re-requestor server IP address: **159.79.81.34**

**MoldUDP64 sessions** for requesting lost messages. UDP port in the above host: **31535** (used in client’s sendto operation)

**Colo customer’s own L3 switches**

Clients: when detecting “too high” message sequence number (in the MoldUDP64 packet header received in the IP Multicast Prod A flow): send a MoldUDP64 request packet over the Prod A request session, specifying the message sequence numbers being requested.

**Nasdaq Site A**

**SE -Site A**

(*) Premium Colo is detailed in a separate document

Clients: when detecting “too high” message sequence number (in the MoldUDP64 packet header received in the IP Multicast Prod C flow): send a MoldUDP64 request packet over the Prod C request session, specifying the message sequence numbers being requested.
This page shows INET Nordic ITCH NLS for Production - Multicast flow applicable to Stockholm Co-location (Site A only) for 10G. (1G on previous page). Different IP addresses compared to 1G. “C” is here provisioned as secondary for 10G Co-location customers.
This page shows INET Nordic ITCH NLS for Test - Multicast flow applicable to Stockholm Co-location (Site A only) for 1G. Both the “A” and “B” flows are the same as earlier described for Test.

**SE -Site A**

**ITCH server, Primary**

- **Test A-Ls**
  - IP address for ITCH Multicast Test A host (source IP address in the Multicast packets): 159.79.85.90
  - Multicast group Test A-Ls:
    - Destination UDP port: 31050
    - Destination IP address (i.e. Multicast address): 233.74.125.50

- **Re-request server**
  - Req Test A-Ls

**ITCH server, Secondary**

- **Test B-Ls**
  - IP address for ITCH Multicast Test B host: 159.79.85.122
  - Multicast group Test B-Ls:
    - Destination UDP port: 31051
    - Destination IP address (i.e. Multicast address): 233.74.125.51

- **Re-request server**
  - Req Test B-Ls

---

**MoldUDP64 sessions (bidirectional over UDP) for requesting lost Test A-Ls messages:**

- Genium INET Nordics UDP port: 31550 (used in client´s sendto operation)

---

**Nasdaq Site A network, with topology for 1G Colo**

- Colo customer´s own L3 switches

---

**Clients:** when detecting “too high” message sequence number (in the MoldUDP64 packet header received in the IP Multicast Test A flow): send a MoldUDP64 request packet over the Test A request session, specifying the message sequence numbers being requested.

- Secondary Re-requestor server IP address: 159.79.87.239 (subnet: 159.79.86.0/23)

---

**MoldUDP64 sessions for requesting lost messages. UDP port in the above host: 31651 (used in client´s sendto operation)**

- IP address for ITCH Multicast Test B host: 159.79.85.122

---

**Clients:** when detecting “too high” message sequence number (in the MoldUDP64 packet header received in the IP Multicast Test B flow): send a MoldUDP64 request packet over the Test B 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 UDP checksum
- Invalid UDP header
- Bad data length
- Overflow (receive UDP 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.

Command `netstat -g` or `ip maddr` gives information on what Multicast groups have been joined.

`netstat -un` or `ss -u` shows the IP mapping between the local IP address and the IP Multicast group with associated UDP port (the IP Multicast group is here regarded as the remote IP address).
Revision History

• **Version 1.0.** First version.