Overload protection in FIXEdge

Overview

Starting from FIXEdge v.5.12.0 there was added a business protection mechanism regulating incoming and outgoing traffic to optimize memory consumption and to prevent system from overloading. The mechanism includes monitoring of throughput and of number of incoming messages received during a session as well as outgoing queue monitoring for slow consumers detection.

Features

- FIXEdge takes measurements to protect itself from higher throughput that may risk of system overloading. Such incidents (i.e. of throughput higher than expected) are monitored and reported. It is essential that the incoming message flow will not exceed substantially the maximum daily expected number of messages from each inbound connection. It is possible to set:
    - The maximum number of messages per second a FIX session can send to FIXEdge
    - The maximum number of messages per day a FIX session can send to FIXEdge
- FIXEdge protects itself from slow consumers by monitoring and reporting the sessions with larger outgoing message queue. FIXEdge provides monitoring and alerting functionality for the outbound queue in order to provide the support team with advanced notice of any possible threat to the system
- In case either threshold is reached, the FIX session will be logged out manually in order to prevent further messages from being sent.

Management of Incoming queue

There is no an inbound queue in FIXEdge/FIX Antenna. A new message is read from TCP buffer after previous one is processed. In other words, a number of waiting messages are limited by TCP buffer size and controlled by the Operating System.

There were added the following parameters in FIXEdge.properties file defined on session level:

1. **IncomingMessagesLimit** – a limit for messages received during a session. The session-level messages are counted as well as application level messages.
2. **IncomingThroughputLimit** – a limit for incoming messages throughput.

Default value for both parameters is '0' - unlimited, no events will be created on business level in such case. Both parameters are available for registered and unregistered sessions.

Management of Outgoing queue

Slow consumers can cause excessive memory consumption, therefore there was added new session parameter in FIXEdge.properties file to control outgoing queue:

1. **OutgoingQueueSize** – a limit for outgoing queue.

Default value for the parameter is '0' - unlimited, no events will be created on business level in such case. Parameter is available for registered and unregistered sessions.

Event Handling

An event **OnNofificationEvent** rises within FIXEdge BL each time when described thresholds are reached.

The event can be handled in BL, for example with javascript or FIXEdge can create an email to be sent to Support Team via SMTP Adaptor. All events are also logged into FIXEdge.log.

Real-time values of the parameters above can be got in a response to **SessionStat** request sent via admin-session to FIXEdge.

Slow Consumer Testing Example

Sessions configuration in FIXEdge.properties file:

FIXEdge.properties

```
# sessions configuration. POOL and LOOP sessions are emulating slow consumer for testing.
FixLayer.FixEngine.Sessions = TEST, LOOP, POOL

FixLayer.FixEngine.Session.LOOP.Version = FIX44
FixLayer.FixEngine.Session.LOOP.StorageType = persistent
FixLayer.FixEngine.Session.LOOP.Role = Acceptor
FixLayer.FixEngine.Session.LOOP.SenderCompID = FIXEDGE
FixLayer.FixEngine.Session.LOOP.TargetCompID = LOOP
FixLayer.FixEngine.Session.LOOP.RecreateOnLogout = true
FixLayer.FixEngine.Session.LOOP.ForceSeqNumReset = true
FixLayer.FixEngine.Session.LOOP.HandleSeqNumAtLogon = false
# setup Outgoing Queue Size limit
FixLayer.FixEngine.Session.LOOP.OutgoingQueueSize = 10

FixLayer.FixEngine.Session.POOL.Version = FIX44
FixLayer.FixEngine.Session.POOL.StorageType = persistent
FixLayer.FixEngine.Session.POOL.Role = Initiator
FixLayer.FixEngine.Session.POOL.Host = 127.0.0.1
FixLayer.FixEngine.Session.POOL.Port = 8901
FixLayer.FixEngine.Session.POOL.HBI = 30
FixLayer.FixEngine.Session.POOL.SenderCompID = LOOP
FixLayer.FixEngine.Session.POOL.TargetCompID = FIXEDGE
FixLayer.FixEngine.Session.POOL.RecreateOnLogout = true
FixLayer.FixEngine.Session.POOL.ForceSeqNumReset = 2
FixLayer.FixEngine.Session.POOL.HandleSeqNumAtLogon = false

FixLayer.FixEngine.Session.TEST.Version = FIX44
FixLayer.FixEngine.Session.TEST.StorageType = persistent
FixLayer.FixEngine.Session.TEST.Role = Acceptor
FixLayer.FixEngine.Session.TEST.SenderCompID = FIXEDGE
FixLayer.FixEngine.Session.TEST.TargetCompID = TEST
FixLayer.FixEngine.Session.TEST.RecreateOnLogout = true
FixLayer.FixEngine.Session.TEST.IntradayLogoutTolerance = true
FixLayer.FixEngine.Session.TEST.ForceSeqNumReset = false
FixLayer.FixEngine.Session.TEST.HandleSeqNumAtLogon = false
FixLayer.FixEngine.Session.TEST.HiddenLogonCredentials = true
# setup Outgoing Queue Size limit
FixLayer.FixEngine.Session.TEST.IncomingMessagesLimit = 10000
FixLayer.FixEngine.Session.TEST.IncomingThroughputLimit = 400
```

Configure and enable SMTP TA in FIXEdge.properties file:

FIXEdge.properties

```
TransportLayer.TransportAdapters = TransportLayer.SmtpTA
```

Setup routing rules and slow consumer emulation in BL_Config.xml:

```
        <!-- ================================================================================== -->
        <!-- ===================== Rules for Slow Consumers testing ============================-->
        <!-- ================================================================================== -->

    <Rule>
        <Source>
            <FixSession SenderCompID="TEST" TargetCompID="NEPTUNECERT"/>
     </Source>
     <Condition>
         <MatchField Field="35" Value="D" />
     </Condition>
     <Action>
         <Send Name="LOOP"/>
     </Action>
 </Rule>

 <Rule>
     <Source Name="POOL"/>
     <Condition>
         <MatchField Field="35" Value="D" />
     </Condition>
     <Action>
                <!-- Send back to TEST session with delay -->
         <Script Language="JavaScript" FileName ="FIXEdge1/conf/sleep.js"/>
         <Send Name="TEST"/>
     </Action>
 </Rule>

        <!-- ===================== Process BL Events ================================-->
        <OnNotificationEvent>
          <Source>
            <FixSession SenderCompID=".*" TargetCompID=".*" />
          </Source>
          <Condition>
            <EqualField Field="35" Value="C"/>
            <MatchField Field="147" Value="\[WARN\].*Session limit watch:.*is reached!"/>
          </Condition>
          <Action>
               <!--<DisconnectSession SenderCompID="NEPTUNECERT" TargetCompID="LOOP" Reason="Disconnect due to
limit" />-->
               <Send>
                   <Client Name="TestSMTPClient"/>
                   </Send>
          </Action>
        </OnNotificationEvent>
```

Slow consumer delays are emulated by sleep.js:

sleep.js

```
function sleep(ms) {
  ms += new Date().getTime();
  while (new Date() < ms){}
}
sleep(100);
```

Steps to reproduce

1. Start FIXEdge with described configuration;
2. Connect 'TEST' (TEST/FIXEdge) session to FIXEdge;
3. Send bunch of 10 000 New Order Singles (35=D) messages with max speed to session 'TEST'
4. Session FIXEDGE/LOOP emulates slow consumer and process only 10 messages per second.
5. When the limits are reached FIXEdge send emails via SMTP

Recommendations

Actual parameters values are defined for each system individually and may vary depending on system requirements and current system load.

The value of **IncomingMessagesLimit** = 10 000 from the example above is a very low limit for a real message processing and was used to demonstrate FIXEdge protection mechanism only. General recommendation for **IncomingMessagesLimit** is > 10000000 (10+ million) per session.

Recommended value for **IncomingThroughputLimit** is 10 000 msg/s per session. Higher throughput for a session can affect performance of other sessions.

OutgoingQueueSize more that 100 should warn of an abnormal situation. Normal outgoing queue size is ~ 0-5.