This brief guide demonstrates how to use the Solace DotNet API to implement a Solace Consumer for SWIM. The basic steps for connecting to a SWIM Solace Queue are to (1) Create a Context, (2) Create a Session, and (3) Create a Flow. There are also three EventHandlers that need to be defined, a SessionEventHandeler that gets raised when Session events occurs, a FlowEventHandeler that gets raised when Flow events occur, and a MessageEventHandeler that gets raised when a message is received. Both the Session and Flow call for a MessageEventHandeler but you only need to provide it to the Flow because SWIM uses Durable Queues which only provides messages over a Flow and not a Session.

Basic SCDS SOLACE Dotnet SWIM Consumer Developrs Guide

**1) Define the Context Properties and create the Context**

ContextFactoryProperties contextFactoryProperties = new ContextFactoryProperties();

ContextProperties contextProps = new ContextProperties();

ContextFactory.Instance.Init(contextFactoryProperties);

IContext iContext = ContextFactory.Instance.CreateContext(contextProps, null);

**2) Create Cert Collection, define Session Properties, create a Session, and connect to it**

Drag this cert file into a directory and add it to a certificate collection 

X509CertificateCollection certColl = new X509CertificateCollection();

certColl.Add(new X509Certificate(@"\path\to\cert\scds.cert"));

SessionProperties sessionProps = new SessionProperties()

{

Host = connectionHost,

UserName = connectionCredentials.UserName,

Password = connectionCredentials.Password,

VPNName = sessionVPN,

SSLTrustStore = certColl,

};

ISession iSession = iContext.CreateSession(

sessionProps,

null,

handelSessionEvent);

this.iSession.Connect();

**3) Define the Queue to consume from**

IQueue iQueue = ContextFactory.Instance.CreateQueue(queue);

**4) Define the Flow Properties and create the Flow**

FlowProperties flowProps = new FlowProperties()

{

FlowStartState = true,

AckMode = MessageAckMode.AutoAck

};

IFlow iFlow = iSession.CreateFlow(

flowProps,

this.iQueue,

null,

handelMessageEvent,

handelFlowEvent);

At this point the consumer will be connected and receiving messages which are exposed through the defined MessageEventHandeler. An example is provided below that saves the received XML Message to a file. For high message rates services like STDDS TAIS, the processing of message should be done in one or more separate threads.

private static void handelMessageEvent(Object source, MessageEventArgs args)

{

XmlDocument swimMessage = new XmlDocument();

if (args.Message.XmlContent != null)

{

using (MemoryStream xmlStream = new MemoryStream(args.Message.XmlContent))

swimMessage.Load(xmlStream);

File.AppendAllText(args.Message.ADMessageId + ".xml", swimMessage.OuterXml);

}

}