Tom's Blog -

ZeroMQ using Java

MARKDOWN Still have in my mind using ZeroMQ as Transport-Layer for my thrift-based communication between Server and Server and Server and Client. Therefore I took a deep look into JZMQ which is the JAVA-JNI Mapping for ZeroMQ. At the beginning I thought that Curve based encryptions wasn't built in yet, but it was only Certificate based authentification that wasn't implemented,yet. And even though I actually don't need to use it, I gave it a try to port it to Java. This resulted in another [pullrequest](https://github.com/zeromq/jzmq/pull/445) , let's see if that will be accepted, as this is something else than the former examples that I fixed for czmq. (**EDIT**: *Just saw that travis is complaining about the jUnit-tests I wrote. One seems to keep blocking on OpenJDK6! Argh* **)** Using this I could port all security examples to java. Here as example the last certificate based authentication: ```java package org.tt.zmq.security; import java.nio.charset.Charset; import org.zeromq.ZAuth; import org.zeromq.ZCert; import org.zeromq.ZContext; import org.zeromq.ZMQ; //The Ironhouse Pattern // //Security doesn't get any stronger than this. An attacker is going to //have to break into your systems to see data before/after encryption. public class Ironhouse { private static final String CERTIFICATE_FOLDER=".curve"; public static void main(String[] args) { // Create context ZContext ctx = new ZContext(); // Start an authentication engine for this context. This engine // allows or denies incoming connections (talking to the libzmq // core over a protocol called ZAP). ZAuth auth = new ZAuth(ctx); // Get some indication of what the authenticator is deciding auth.setVerbose(true); // Whitelist our address; any other address will be rejected auth.allow("127.0.0.1"); // Tell authenticator to use the certificate store in .curve auth.configureCurve(CERTIFICATE_FOLDER); // We'll generate a new client certificate and save the public part // in the certificate store (in practice this would be done by hand // or some out-of-band process). ZCert client_cert = new ZCert(); client_cert.setMeta("name", "Client test certificate"); client_cert.savePublic(CERTIFICATE_FOLDER+"/testcert.pub"); ZCert server_cert = new ZCert(); // Create and bind server socket ZMQ.Socket server = ctx.createSocket(ZMQ.PUSH); server.setZAPDomain("global".getBytes()); server.setCurveServer(true); server.setCurvePublicKey(server_cert.getPublicKey()); server.setCurveSecretKey(server_cert.getSecretKey()); server.bind("tcp://*:9000"); // Create and connect client socket ZMQ.Socket client = ctx.createSocket(ZMQ.PULL); client.setCurvePublicKey(client_cert.getPublicKey()); client.setCurveSecretKey(client_cert.getSecretKey()); client.setCurveServerKey(server_cert.getPublicKey()); client.connect("tcp://127.0.0.1:9000"); // Send a single message from server to client server.send("Hello"); String message = client.recvStr(0,Charset.defaultCharset()); if (message.equals("Hello")) { System.out.println("Ironhouse test OK"); } ctx.close(); } } ```

Published by Thomas Trocha
Thu Jan 05 03:59:00 CET 2017
http://thomas.trocha.com:80/pebble/zeromq-using-java