#include jNetPcap API;

// template for objects that represent IPv4 packets

Class Packet

{

// variables

String srcIp;

String dstIp;

String dstPort;

String ipHeader;

String udpPayload;

String completePacket;

Int packetSize;

// constructor, for received packet

Void packet(String completePacket)

{

1. Set this.completePacket = completePacket;

2. Set this.packetSize = packetSizeCalc(completePacket);

}

// constructor, for packet to transmit

Void packet(String srcIp, String dstIp, String dstPort)

{

1. Set this.srcIp = srcIp;

2. Set this.dstIp = dstIp;

3. Set this.dstPort = dstPort;

4. Set this.ipHeader = ipHeaderMaker(srcIp, dstIp);

5. Set this.udpPayload = udpPayloadMaker(dstPort);

6. Set this.completePacket = combiner(ipHeader, udpPayload);

7. Set this.packetSize = packetSizeCalc(completePacket);

}

// create IP header

String ipHeaderMaker(String srcIp, String dstIp)

{

1. Return jNetPcap.header(IP4, srcIp, dstIp);

}

// create UDP payload

String udpPayloadMaker(dstPort)

{

1. String udpPayloadTemplate = “... status request ...”;

2. Int portOffset = 4;

3. Overwrite contents of udpPayloadTemplate at portOffset with dstPort;

4. Return udpPayloadTemplate;

}

// combine IP header and UDP payload to make a complete packet

String combiner(String ipHeader, String udpPayload)

{

1. Return ipHeader concatenated with udpPayload;

}

// calculate size of packet

int packetSizeCalc(String completePacket)

{

1. Return completePacket.size();

}

String toString()

{

1. Return completePacket;

}

} // end class Packet