NEXT GENERATION NETWORKS

OMNET++ SIMULATION OF AODV AND DSDV PROTOCOLS

AODV (Ad-hoc On-demand Distance Vector) Routing Protocol

The Ad-hoc On-demand Distance Vector (AODV) is a widely used routing protocol in mobile ad-hoc networks (MANETs). AODV enables dynamic, self-starting, multi-hop routing between mobile nodes wishing to establish and maintain an ad-hoc network. It is characterized by its on-demand approach, where routes are created as needed, and its distance vector methodology, where the distance to the destination is the primary metric used for route selection.

Key Features of AODV

1. **On-Demand Route Discovery**:
    - Routes are established only when they are needed to reduce overhead.
    - This minimizes the control traffic compared to proactive protocols, which maintain complete routing tables at all times.
2. **Loop-Free Routing**:
    - AODV ensures loop-free routes by using sequence numbers. Each node maintains a monotonically increasing sequence number to keep track of the freshness of routes.
    - Sequence numbers help in avoiding routing loops and in providing up-to-date routing information.
3. **Route Maintenance**:
    - AODV maintains routes as long as they are needed. When a route is no longer required or a node moves out of range, the route is removed.
    - Broken links are quickly detected using periodic HELLO messages or by other nodes informing of link failures.
4. **Scalability**:
    - AODV scales well to large networks due to its on-demand nature. The routing table size and control overhead are proportional to the number of active routes rather than the total number of nodes.
5. **Support for Unicast and Multicast**:
    - AODV can handle both unicast (one-to-one) and multicast (one-to-many) routing, making it versatile for various communication scenarios in ad-hoc networks.

How AODV Works

AODV operates through two primary mechanisms: route discovery and route maintenance.

Route Discovery

1. **Route Request (RREQ)**:

- When a source node wants to communicate with a destination node for which it does not have a valid route, it broadcasts a Route Request (RREQ) packet to its neighbors.
- The RREQ packet includes the following information:
  - Source IP address and current sequence number.
  - Destination IP address and the last known sequence number.
  - Broadcast ID (unique identifier for the RREQ initiated by the source).
  - Time-to-Live (TTL) indicating how many hops the RREQ can travel.

2. **Propagation of RREQ**:
   - Intermediate nodes receiving the RREQ check if they have a fresh route to the destination by comparing the sequence number in the RREQ with the sequence number of their existing routes.
   - If a fresh route is available, the intermediate node sends a Route Reply (RREP) back to the source.
   - If not, the node rebroadcasts the RREQ, updating its own routing table to include the reverse path to the source.
3. **Route Reply (RREP)**:
   - When the destination node or an intermediate node with a fresh route to the destination receives the RREQ, it generates an RREP.
   - The RREP is unicast back to the source node along the reverse path established during the RREQ broadcast.
   - The RREP includes the destination IP address, the sequence number, the hop count to the destination, and a lifetime indicating how long the route is valid.
4. **Establishment of Route**:
   - The source node receives the RREP and updates its routing table with the route to the destination.
   - The route is then ready to be used for data transmission.

Route Maintenance

1. **Route Error (RERR)**:
   - If a link failure is detected, the detecting node generates a Route Error (RERR) packet.
   - The RERR is sent to all affected nodes that were using the broken link, informing them of the link failure.
2. **Route Expiry**:
   - Routes are maintained in the routing table with a lifetime indicating how long they are valid.
   - If no data packets are sent over a route within its lifetime, the route is considered expired and is deleted from the routing table.
3. **Periodic HELLO Messages**:
   - Nodes send periodic HELLO messages to maintain local connectivity information.
   - Failure to receive a HELLO message from a neighbor within a certain interval indicates a link breakage.

Advantages and Disadvantages of AODV

Advantages

- **On-Demand Nature**: Reduces unnecessary control traffic by establishing routes only when needed.
- **Dynamic Topology Support**: Efficiently handles the dynamic topology of MANETs where nodes frequently move.
- **Loop-Free Routing**: Sequence numbers ensure that routes are loop-free and the most current.

Disadvantages

- **High Route Discovery Latency**: Initial route discovery can introduce delays, especially in large networks.
- **Route Maintenance Overhead**: Frequent route breaks in highly mobile environments can lead to increased control traffic.
- **Scalability Issues**: While AODV scales well compared to some protocols, performance can degrade with very large network sizes and high mobility.

Use Cases

- **Disaster Recovery**: AODV is ideal for setting up communication networks quickly in disaster-affected areas where infrastructure is unavailable or damaged.
- **Military Networks**: Suitable for military operations requiring secure, ad-hoc communication in rapidly changing environments.
- **Sensor Networks**: Can be used in sensor networks where nodes are deployed in an ad-hoc manner and need efficient, on-demand routing.

AODV's design makes it a robust choice for ad-hoc networks where routes need to be established dynamically and maintained with minimal overhead, making it a vital protocol in scenarios demanding flexibility and quick adaptation to changing network conditions.

CODE

```
#include "inet/routing/aodv/Aodv.h"

#include "inet/common/IProtocolRegistrationListener.h"
#include "inet/common/ModuleAccess.h"
#include "inet/common/ProtocolTag_m.h"
#include "inet/common/packet/Packet.h"
#include "inet/common/stlutils.h"
#include "inet/linklayer/common/InterfaceTag_m.h"
#include "inet/networklayer/common/HopLimitTag_m.h"
#include "inet/networklayer/common/L3AddressResolver.h"
#include "inet/networklayer/common/L3AddressTag_m.h"
#include "inet/networklayer/common/L3Tools.h"
#include "inet/networklayer/ipv4/IcmpHeader.h"
#include "inet/networklayer/ipv4/Ipv4Header_m.h"
#include "inet/networklayer/ipv4/Ipv4Route.h"
#include "inet/transportlayer/common/L4PortTag_m.h"
#include "inet/transportlayer/contract/udp/UdpControlInfo.h"

namespace inet {
namespace aodv {
```

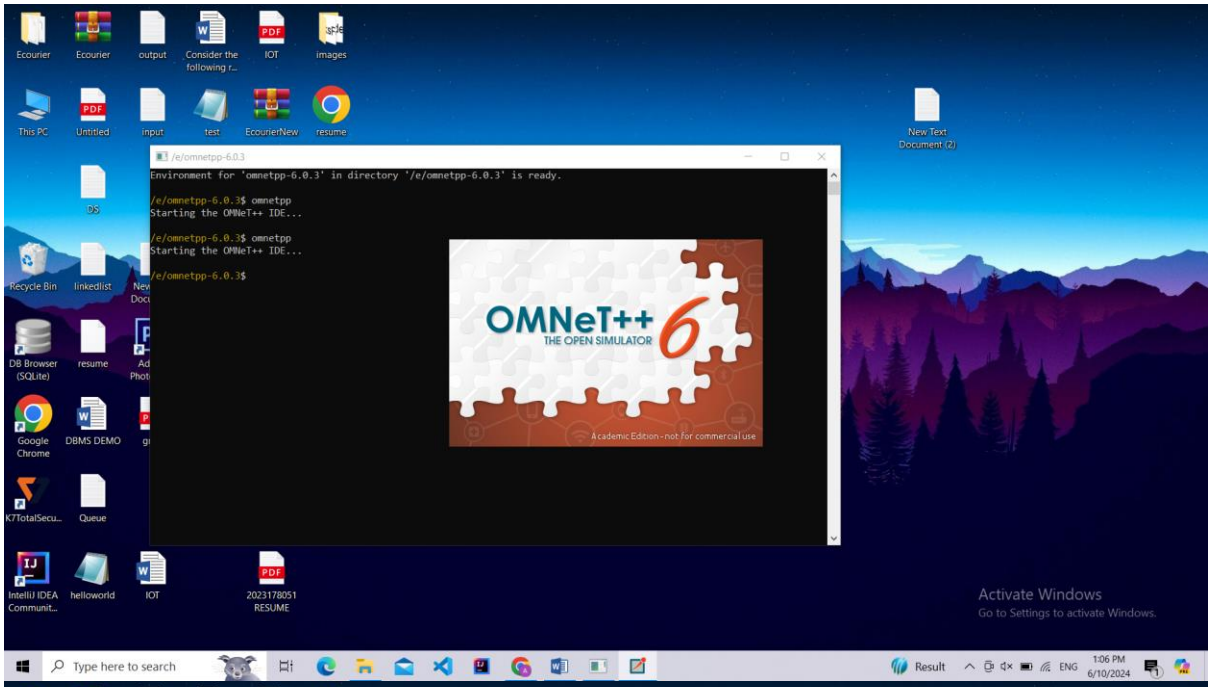
```
Define_Module(Aodv);

const int KIND_DELAYEDSEND = 100;

void Aodv::initialize(int stage)
{
    if (stage == INITSTAGE_ROUTING_PROTOCOLS)
        addressType = getSelfIPAddress().getAddressType(); // needed for
handleStartOperation()

    RoutingProtocolBase::initialize(stage);

    if (stage == INITSTAGE_LOCAL) {
        lastBroadcastTime = SIMTIME_ZERO;
        rebootTime = SIMTIME_ZERO;
        rreqId = sequenceNum = 0;
        rreqCount = rerrCount = 0;
        host = getContainingNode(this);
        routingTable.reference(this, "routingTableModule", true);
        interfaceTable.reference(this, "interfaceTableModule", true);
        networkProtocol.reference(this, "networkProtocolModule", true);

        aodvUDPPort = par("udpPort");
        askGratuitousRREP = par("askGratuitousRREP");
        useHelloMessages = par("useHelloMessages");
        destinationOnlyFlag = par("destinationOnlyFlag");
        activeRouteTimeout = par("activeRouteTimeout");
        helloInterval = par("helloInterval");
        allowedHelloLoss = par("allowedHelloLoss");
        netDiameter = par("netDiameter");
        nodeTraversalTime = par("nodeTraversalTime");
        rerrRatelimit = par("rerrRatelimit");
        rreqRetries = par("rreqRetries");
        rreqRatelimit = par("rreqRatelimit");
        timeoutBuffer = par("timeoutBuffer");
        ttlStart = par("ttlStart");
        ttlIncrement = par("ttlIncrement");
        ttlThreshold = par("ttlThreshold");
        localAddTTL = par("localAddTTL");
        jitterPar = &par("jitter");
        periodicJitter = &par("periodicJitter");

        myRouteTimeout = par("myRouteTimeout");
        deletePeriod = par("deletePeriod");
        blacklistTimeout = par("blacklistTimeout");
        netTraversalTime = par("netTraversalTime");
        nextHopWait = par("nextHopWait");
        pathDiscoveryTime = par("pathDiscoveryTime");
        expungeTimer = new cMessage("ExpungeTimer");
        counterTimer = new cMessage("CounterTimer");
        rrepAckTimer = new cMessage("RrepAckTimer");
        blacklistTimer = new cMessage("BlackListTimer");
        if (useHelloMessages)
            helloMsgTimer = new cMessage("HelloMsgTimer");
    }
    else if (stage == INITSTAGE_ROUTING_PROTOCOLS) {
        networkProtocol->registerHook(0, this);
        host->subscribe(linkBrokenSignal, this);
```

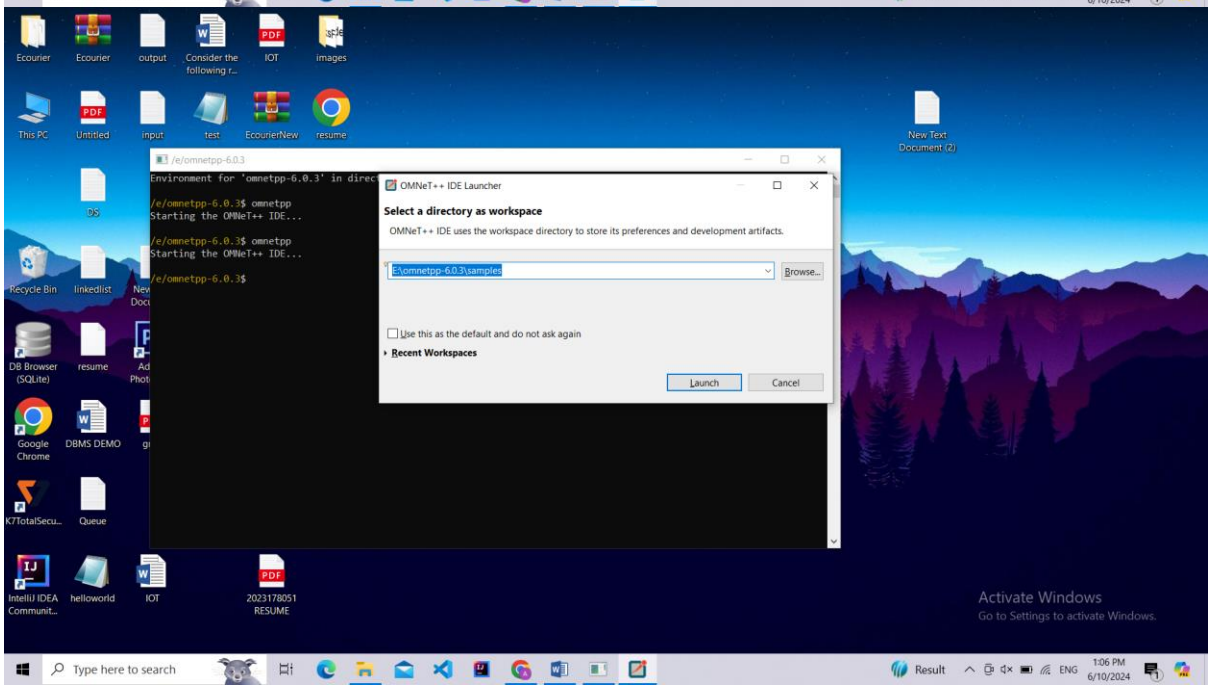
```
        usingIpv6 = (routingTable->getRouterIdAsGeneric().getType() ==
L3Address::IPv6);
    }
}

void Aodv::handleMessageWhenUp(cMessage *msg)
{
    if (msg->isSelfMessage()) {
        if (auto waitForRrep = dynamic_cast<WaitForRrep *>(msg))
            handleWaitForRREP(waitForRrep);
        else if (msg == helloMsgTimer)
            sendHelloMessagesIfNeeded();
        else if (msg == expungeTimer)
            expungeRoutes();
        else if (msg == counterTimer) {
            rreqCount = rerrCount = 0;
            scheduleAfter(1, counterTimer);
        }
        else if (msg == rrepAckTimer)
            handleRREPACKTimer();
        else if (msg == blacklistTimer)
            handleBlackListTimer();
        else if (msg->getKind() == KIND_DELAYEDSEND) {
            auto timer = check_and_cast<PacketHolderMessage *>(msg);
            socket.send(timer->removeOwnedPacket());
            delete timer;
        }
        else
            throw cRuntimeError("Unknown self message");
    }
    else
        socket.processMessage(msg);
}

void Aodv::checkIpVersionAndPacketTypeCompatibility(AodvControlPacketType
packetType)
{
    switch (packetType) {
        case RREQ:
        case RREP:
        case RERR:
        case RREPACK:
            if (usingIpv6)
                throw cRuntimeError("AODV Control Packet arrived with non-IPv6
packet type %d, but AODV configured for IPv6 routing", packetType);
            break;

        case RREQ_IPv6:
        case RREP_IPv6:
        case RERR_IPv6:
        case RREPACK_IPv6:
            if (!usingIpv6)
                throw cRuntimeError("AODV Control Packet arrived with IPv6 packet
type %d, but AODV configured for non-IPv6 routing", packetType);
            break;

        default:
            throw cRuntimeError("AODV Control Packet arrived with undefined packet
type: %d", packetType);
```

```
    }
}

void Aodv::processPacket(Packet *packet)
{
    L3Address sourceAddr = packet->getTag<L3AddressInd>()->getSrcAddress();
    // KLUDGE I added this -1 after TTL decrement has been moved in Ipv4
    unsigned int arrivalPacketTTL = packet->getTag<HopLimitInd>()->getHopLimit() -
1;
    const auto& aodvPacket = packet->popAtFront<AodvControlPacket>();
    // TODO aodvPacket->copyTags(*udpPacket);

    auto packetType = aodvPacket->getPacketType();
    switch (packetType) {
        case RREQ:
        case RREQ_IPv6:
            checkIpVersionAndPacketTypeCompatibility(packetType);
            handleRREQ(CHK(dynamicPtrCast<Rreq>(aodvPacket->dupShared())),
sourceAddr, arrivalPacketTTL);
            delete packet;
            return;

        case RREP:
        case RREP_IPv6:
            checkIpVersionAndPacketTypeCompatibility(packetType);
            handleRREP(CHK(dynamicPtrCast<Rrep>(aodvPacket->dupShared())),
sourceAddr);
            delete packet;
            return;

        case RERR:
        case RERR_IPv6:
            checkIpVersionAndPacketTypeCompatibility(packetType);
            handleRERR(CHK(dynamicPtrCast<const Rerr>(aodvPacket)), sourceAddr);
            delete packet;
            return;

        case RREPACK:
        case RREPACK_IPv6:
            checkIpVersionAndPacketTypeCompatibility(packetType);
            handleRREPACK(CHK(dynamicPtrCast<const RrepAck>(aodvPacket)),
sourceAddr);
            delete packet;
            return;

        default:
            throw cRuntimeError("AODV Control Packet arrived with undefined packet
type: %d", packetType);
    }
}
```

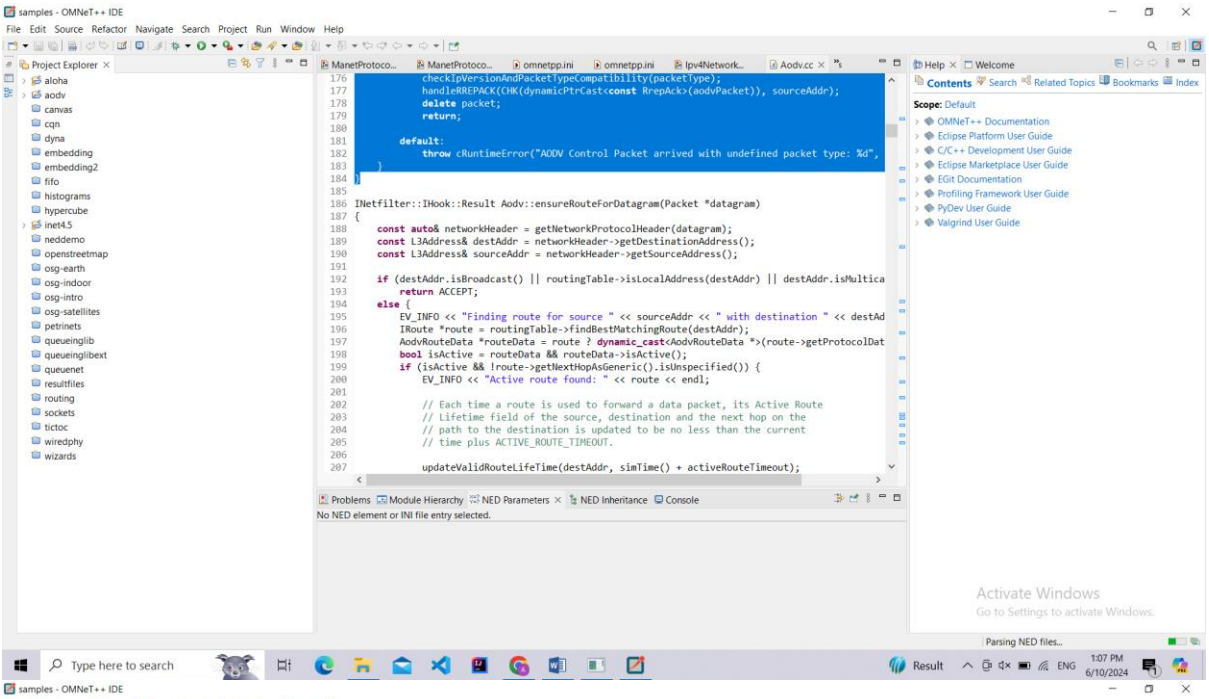
SNAPSHOTS

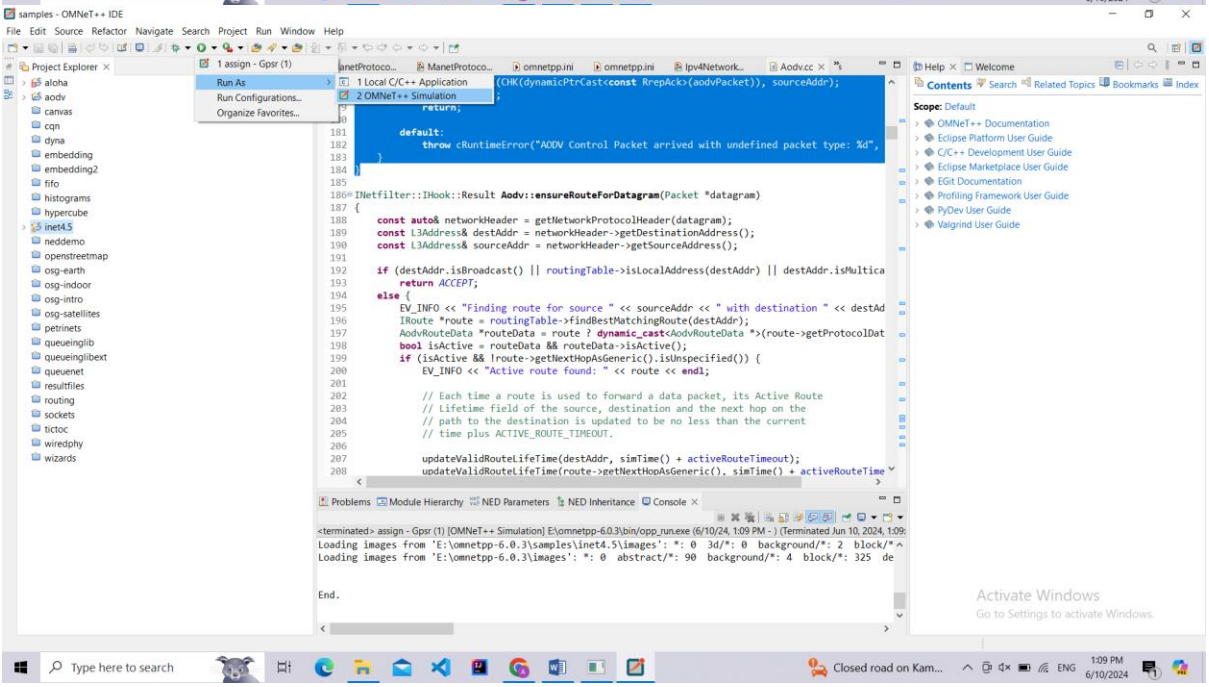
RUNNING OMNET++





RUNNING SIMULATION





DSDV (Destination-Sequenced Distance-Vector) Routing Protocol

The Destination-Sequenced Distance-Vector (DSDV) is a proactive routing protocol designed for mobile ad-hoc networks (MANETs). Unlike on-demand protocols like AODV, DSDV maintains consistent and up-to-date routes to all nodes within the network. It is based on the classical Bellman-Ford algorithm but includes enhancements to ensure loop-free and efficient routing.

Key Features of DSDV

1. **Proactive Routing**:
    - DSDV maintains complete routing information to all possible destinations at all times.
    - Each node maintains a table that lists all available destinations, the number of hops to reach them, and the next hop.
2. **Sequence Numbers**:
    - Each route is tagged with a sequence number to distinguish stale routes from fresh ones.
    - Sequence numbers are generated by the destination nodes, ensuring that the most recent and loop-free routes are maintained.
3. **Periodic Updates**:
    - Nodes periodically transmit updates to their routing tables to maintain the latest route information.
    - These updates can be full or incremental, depending on the extent of the changes in the network topology.
4. **Loop-Free**:
    - The use of sequence numbers helps prevent routing loops by ensuring that each node adopts the most recent route information.
5. **Scalability**:
    - Suitable for small to moderately large networks due to its proactive nature, which incurs constant overhead regardless of traffic demand.

How DSDV Works

DSDV operates through a systematic process of route table maintenance, involving periodic and triggered updates.

Route Table Maintenance

1. **Routing Table**:
    - Each node maintains a routing table containing:
        - The destination address.
        - The number of hops to the destination.
        - The next hop address.
        - A sequence number assigned by the destination.
        - An install time indicating when the route was last updated.
2. **Sequence Numbers**:
    - Each entry in the routing table includes a sequence number that ensures the freshness of the route.

    - The sequence number is incremented periodically by the destination node, and nodes adopt routes with the highest sequence numbers.
3. **Periodic Updates**:
    - Nodes periodically broadcast routing table updates to ensure all nodes have up-to-date information.
    - Full updates include the entire routing table, whereas incremental updates include only the changes since the last full update.
    - Periodic updates are designed to reduce control overhead while keeping routing information current.
4. **Triggered Updates**:
    - When a significant topology change occurs (e.g., a link break), an immediate triggered update is sent.
    - This ensures that broken routes are quickly removed, and new routes are propagated throughout the network.
5. **Routing Metrics**:
    - DSDV uses the hop count as the primary metric for route selection.
    - Nodes select the route with the least number of hops to the destination.

Route Advertisement and Update Process

1. **Advertising Routes**:
    - Each node periodically broadcasts its routing table, including sequence numbers and hop counts.
    - Neighbors receiving the advertisement update their tables if the received route is fresher (i.e., has a higher sequence number) or has a shorter hop count.
2. **Route Aggregation**:
    - To minimize the size of updates, DSDV supports route aggregation, where multiple routes can be combined into a single advertisement if they share common paths.
3. **Route Expiry**:
    - Routes in the table have a validity period. If a route is not updated within this period, it is considered stale and removed.

Advantages and Disadvantages of DSDV

Advantages

- **Loop-Free**: Sequence numbers ensure that routing loops are avoided, providing reliable routing information.
- **Consistent Routes**: Maintains up-to-date routes to all destinations, reducing the delay in route establishment compared to on-demand protocols.
- **Quick Adaptation to Changes**: Frequent updates allow quick adaptation to network topology changes, maintaining route accuracy.

Disadvantages

- **High Overhead**: Periodic updates generate constant overhead, which can become significant in large networks with high mobility.
- **Scalability Issues**: Not well-suited for very large networks due to the increased size of routing tables and the overhead of maintaining them.
- **Redundant Information**: Regularly updated tables can contain redundant information, leading to unnecessary transmissions.

Use Cases

- **Small to Medium-Sized Networks**: Suitable for networks where the topology does not change frequently and where constant route availability is essential.
- **Static or Low-Mobility Networks**: Ideal for networks with low mobility where the overhead of periodic updates is manageable.
- **Sensor Networks**: Can be used in static sensor networks where nodes are deployed in a fixed topology, and routes are required to be consistent and reliable.

DSDV is a robust and reliable choice for scenarios where network topology is relatively stable, and the overhead of maintaining complete routing information is justified by the need for consistent and quick route availability. Its proactive nature and use of sequence numbers make it a valuable protocol for maintaining efficient and loop-free routing in ad-hoc networks.

Summary

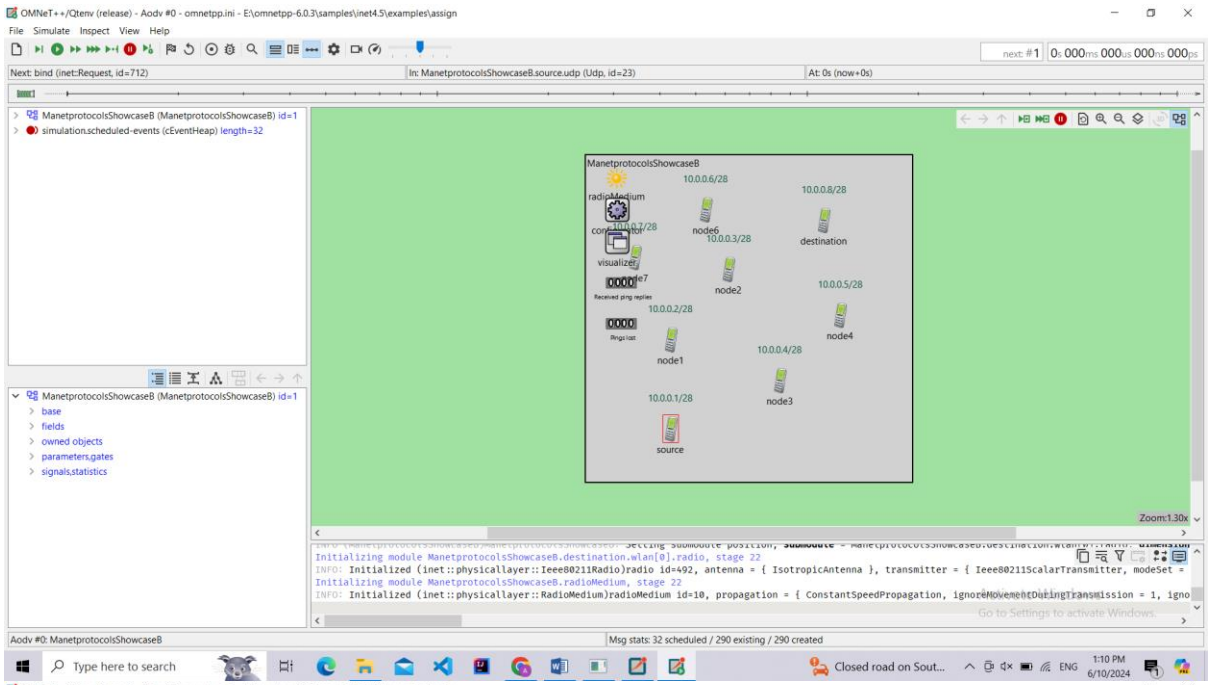
DSDV provides a proactive approach to routing in mobile ad-hoc networks, maintaining consistent and loop-free routes through periodic updates and the use of sequence numbers. While it incurs constant overhead due to regular updates, it ensures that route information is always available, making it suitable for networks where quick route establishment and consistency are critical.

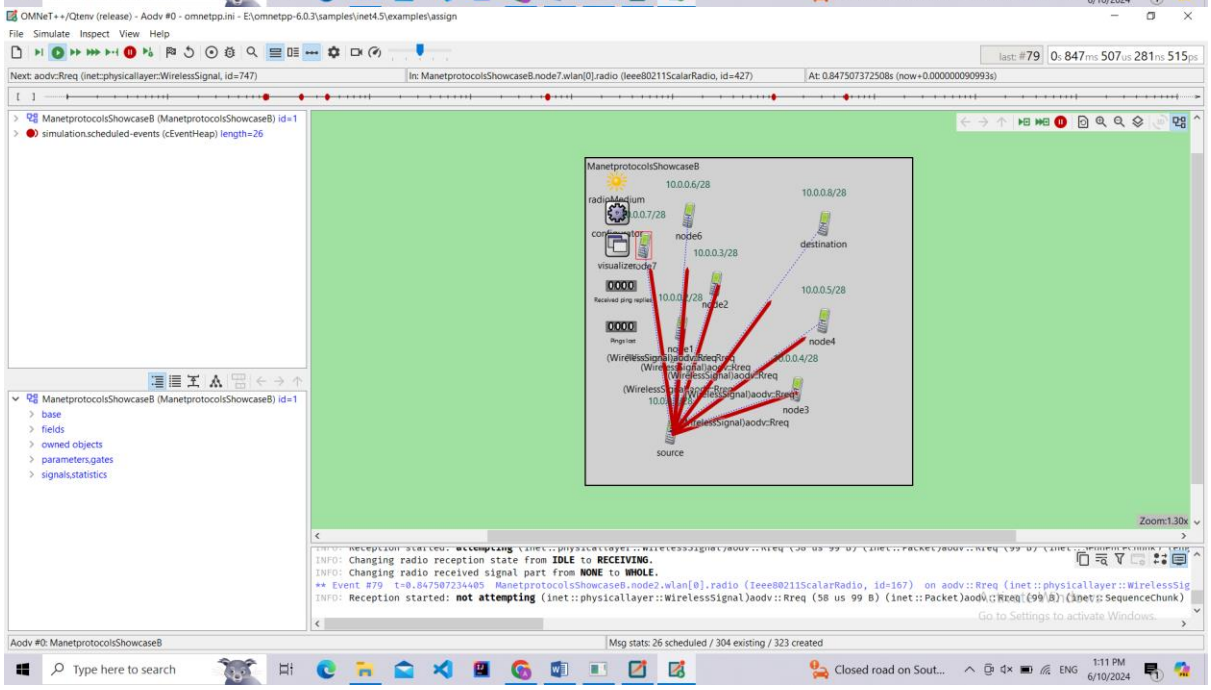
CODE

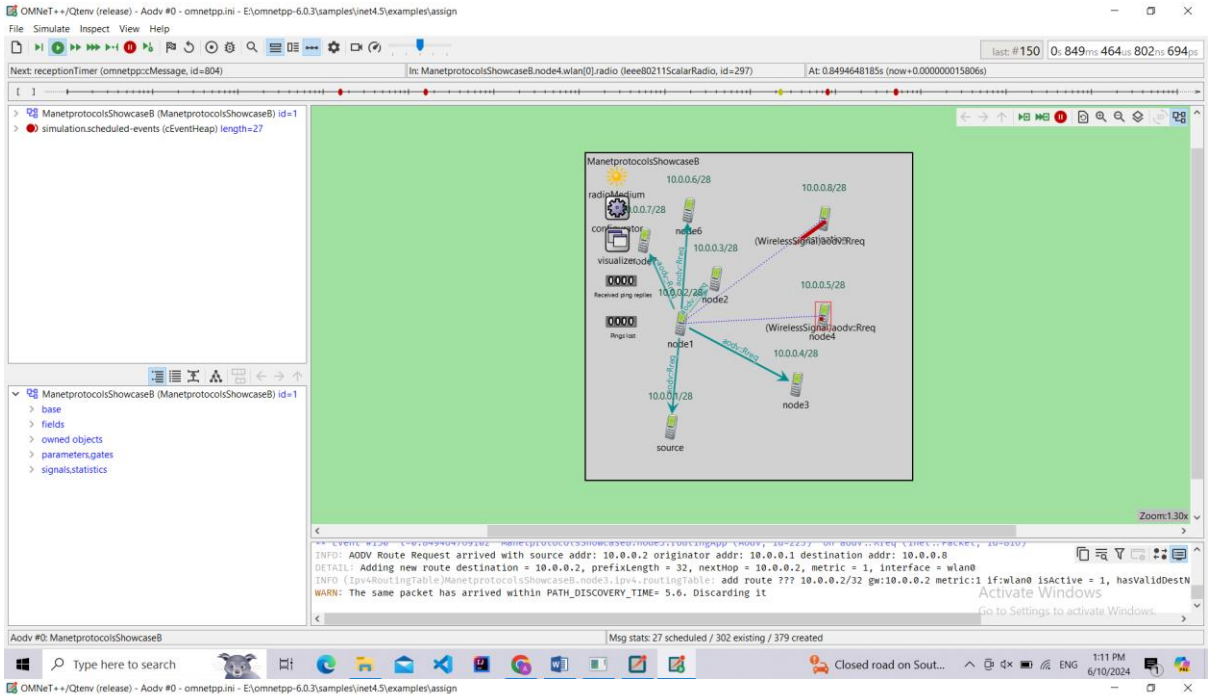
```
//
// Copyright (C) 2014 OpenSim Ltd.
//
// SPDX-License-Identifier: LGPL-3.0-or-later
//

package inet.examples.dsdv;

import inet.common.scenario.ScenarioManager;
import inet.networklayer.configurator.ipv4.Ipv4NetworkConfigurator;
import inet.networklayer.ipv4.RoutingTableRecorder;
import inet.node.dsdv.DsdvRouter;
import inet.physicallayer.wireless.unitdisk.UnitDiskRadioMedium;


network DSDVNetwork
{
   parameters:
```

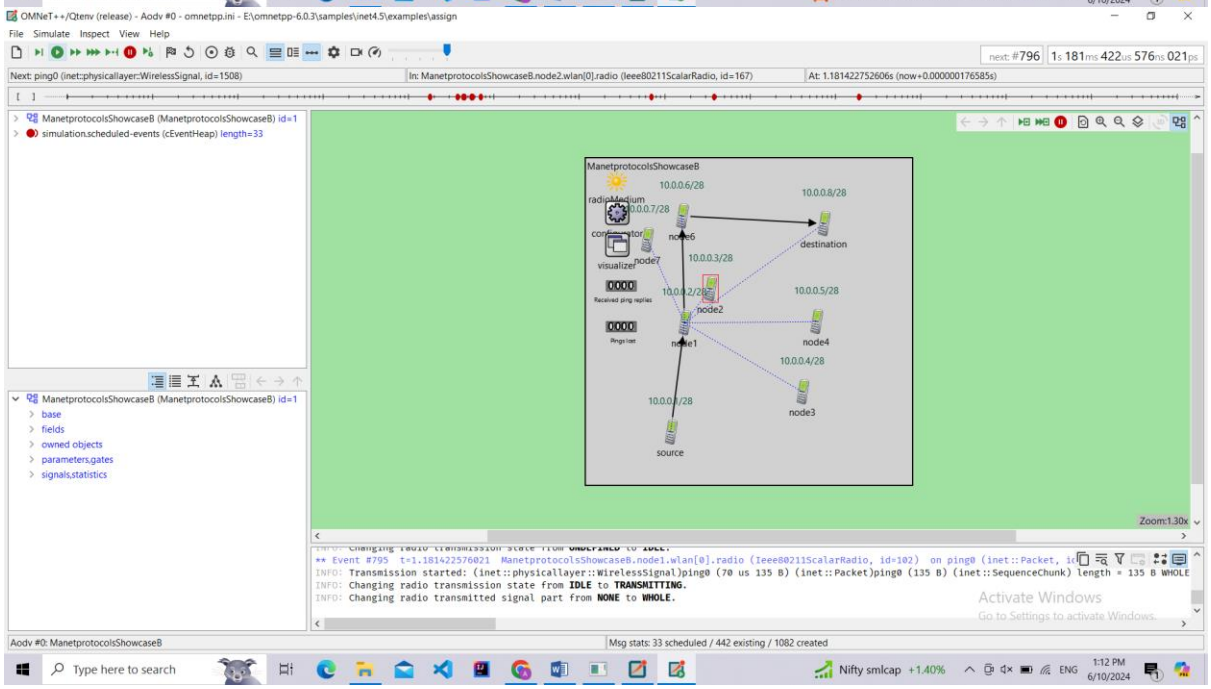
```
        int numHosts;
        @display("bgb=650,650");
    submodules:
        radioMedium: UnitDiskRadioMedium {
            parameters:
                @display("p=100,200;is=s");
        }
        configurator: Ipv4NetworkConfigurator {
            parameters:
                config = xml("<config><interface hosts='*' address='145.236.x.x'
netmask='255.255.0.0'/></config>");
                @display("p=100,100;is=s");
        }
        routingTableRecorder: RoutingTableRecorder {
            parameters:
                @display("p=100,300;is=s");
        }
        scenarioManager: ScenarioManager {
            parameters:
                script = default(xml("<scenario/>"));
                @display("p=100,400;is=s");
        }
        host[numHosts]: DsdvRouter {
            parameters:
                @display("i=device/pocketpc_s;r=,,#707070");
        }
    connections allowunconnected:
}
```

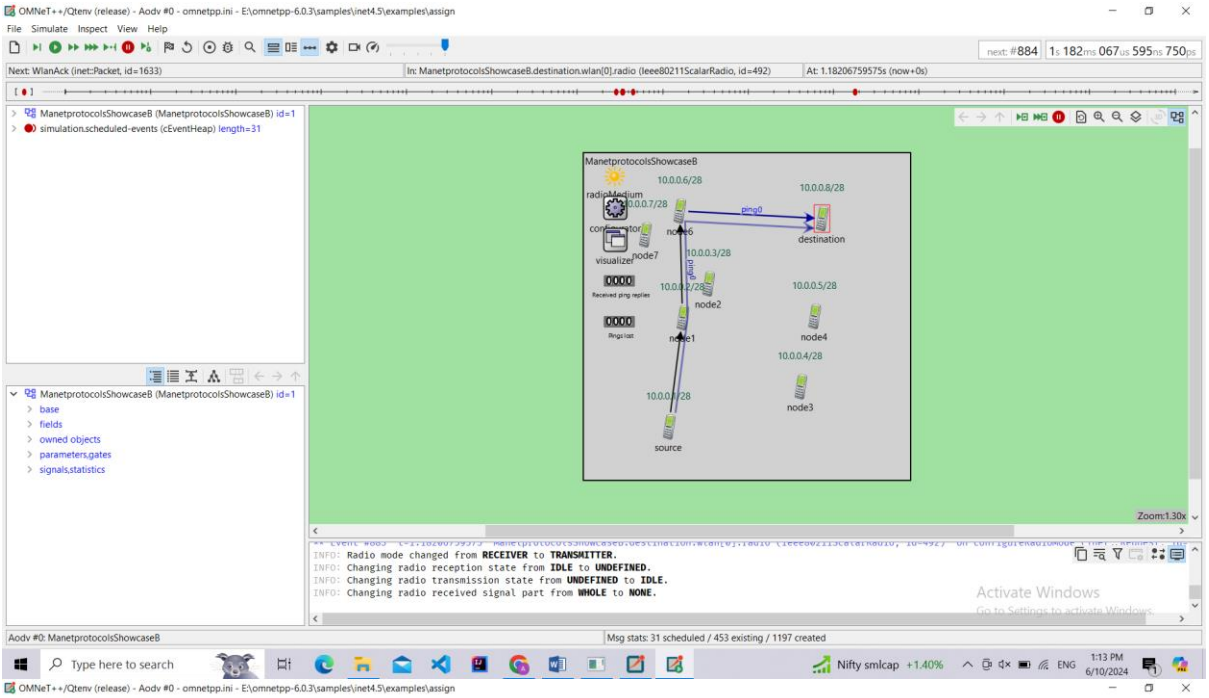
SNAPSHOTS

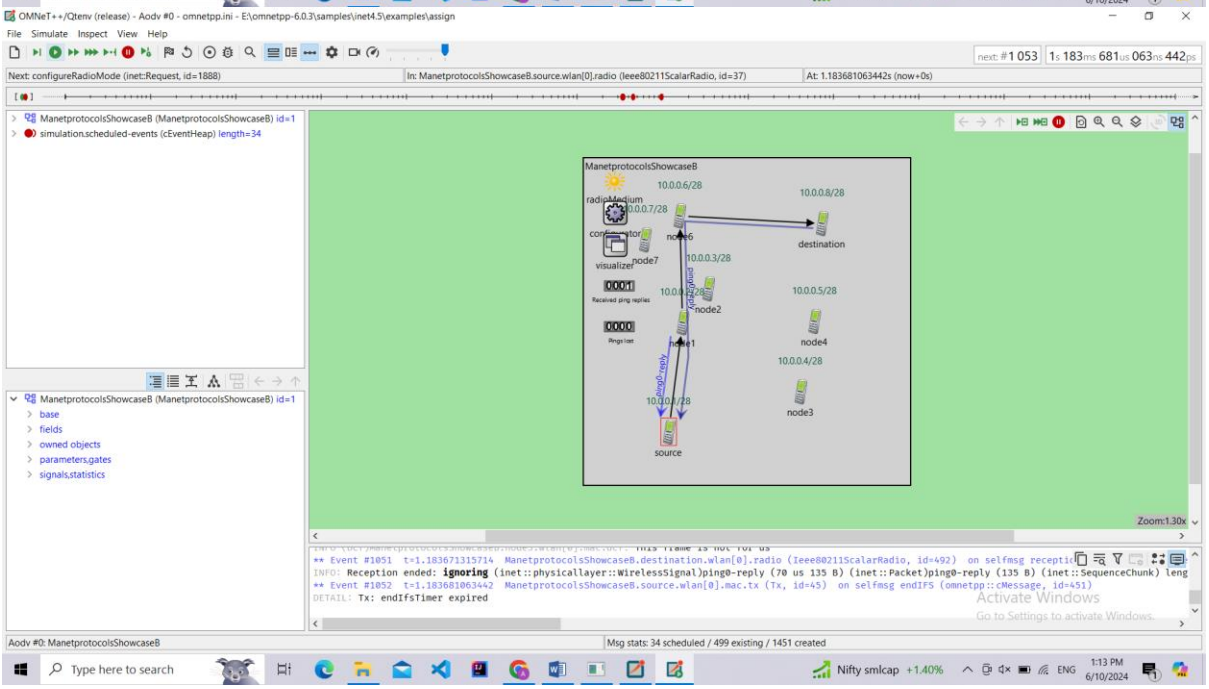

OMNeT++/Qtenv (release) - Aodv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: bind (inet::Request, id=712)
In: ManetprotocolsShowcaseB.source.udp (Udp, id=23)
At: 0s (now+0s)
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=32
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
radioMedium
visualizer
10.0.0.6/28
10.0.0.8/28
10.0.0.7/28
node6
destination
10.0.0.3/28
node7
node2
10.0.0.5/28
10.0.0.2/28
node4
node1
10.0.0.4/28
10.0.0.1/28
node3
source
Zoom:1.30x
Initializing module ManetprotocolsShowcaseB.destination.wlan[0].radio, stage 22
INFO: Initialized (inet::physicallayer::Ieee80211Radio)radio id=492, antenna = { IsotropicAntenna }, transmitter = { Ieee80211ScalarTransmitter, modeSet =
Initializing module ManetprotocolsShowcaseB.radioMedium, stage 22
INFO: Initialized (inet::physicallayer::RadioMedium)radioMedium id=10, propagation = { ConstantSpeedPropagation, ignoreMovementDuringTransmission = 1, igno
Aodv #0: ManetprotocolsShowcaseB
Msg stats: 32 scheduled / 290 existing / 290 created
Type here to search
Closed road on Sout...
ENG
1:10 PM
6/10/2024


OMNeT++/Qtenv (release) - Aodv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: aodv::Rreq (inet::physicallayer::WirelessSignal, id=747)
In: ManetprotocolsShowcaseB.node7.wlan[0].radio (ieee80211ScalarRadio, id=427)
At: 0.847507372508s (now+0.000000090993s)
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=26
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
radioMedium
visualizer
10.0.0.6/28
10.0.0.8/28
10.0.0.7/28
node6
destination
10.0.0.3/28
node7
node2
10.0.0.5/28
node4
node1
10.0.0.4/28
node3
source
Zoom:1.30x
INFO: Changing radio reception state from IDLE to RECEIVING.
INFO: Changing radio received signal part from NONE to WHOLE.
** Event #79 t=0.847507234405 ManetprotocolsShowcaseB.node2.wlan[0].radio (Ieee80211ScalarRadio, id=167) on aodv::Rreq (inet::physicallayer::WirelessSig
INFO: Reception started: not attempting (inet::physicallayer::WirelessSignal)aodv::Rreq (58 us 99 B) (inet::Packet)aodv::Rreq (99 B) (inet::SequenceChunk)
Aodv #0: ManetprotocolsShowcaseB
Msg stats: 26 scheduled / 304 existing / 323 created
Type here to search
Closed road on Sout...
ENG
1:11 PM
6/10/2024

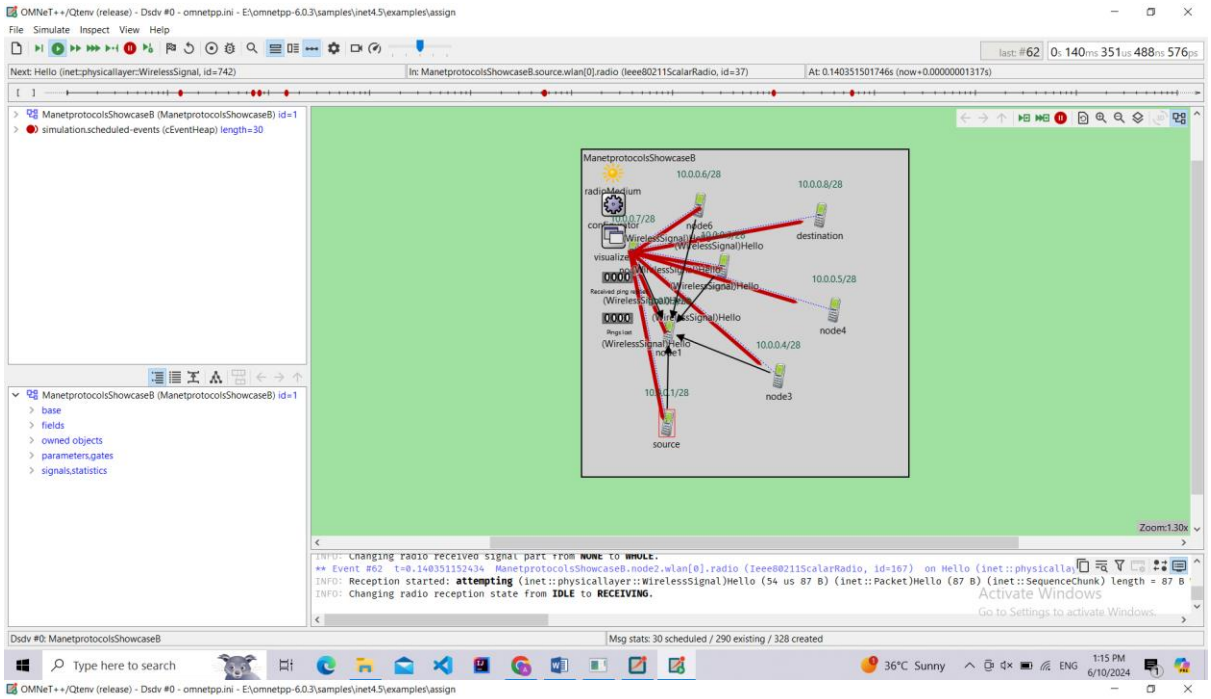

OMNeT++/Qtenv (release) - Aodv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: receptionTimer (omnetpp::cMessage, id=804)
In: ManetprotocolsShowcaseB.node4.wlan[0].radio (ieee80211ScalarRadio, id=297)
At: 0.8494648185s (now+0.000000015806s)
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=27
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
radioMedium
10.0.0.6/28
10.0.0.7/28
10.0.0.8/28
node6
visualizer
10.0.0.3/28
(WirelessSignal)aodv:Rreq
10.0.0.2/28
node2
10.0.0.5/28
node1
node4
10.0.0.4/28
node3
source
Zoom:1.30x
INFO: AODV Route Request arrived with source addr: 10.0.0.2 originator addr: 10.0.0.1 destination addr: 10.0.0.8
DETAIL: Adding new route destination = 10.0.0.2, prefixLength = 32, nextHop = 10.0.0.2, metric = 1, interface = wlan0
WARN: The same packet has arrived within PATH_DISCOVERY_TIME= 5.6. Discarding it
Aodv #0: ManetprotocolsShowcaseB
Msg stats: 27 scheduled / 302 existing / 379 created
Type here to search
Closed road on Sout...
ENG
1:11 PM
6/10/2024

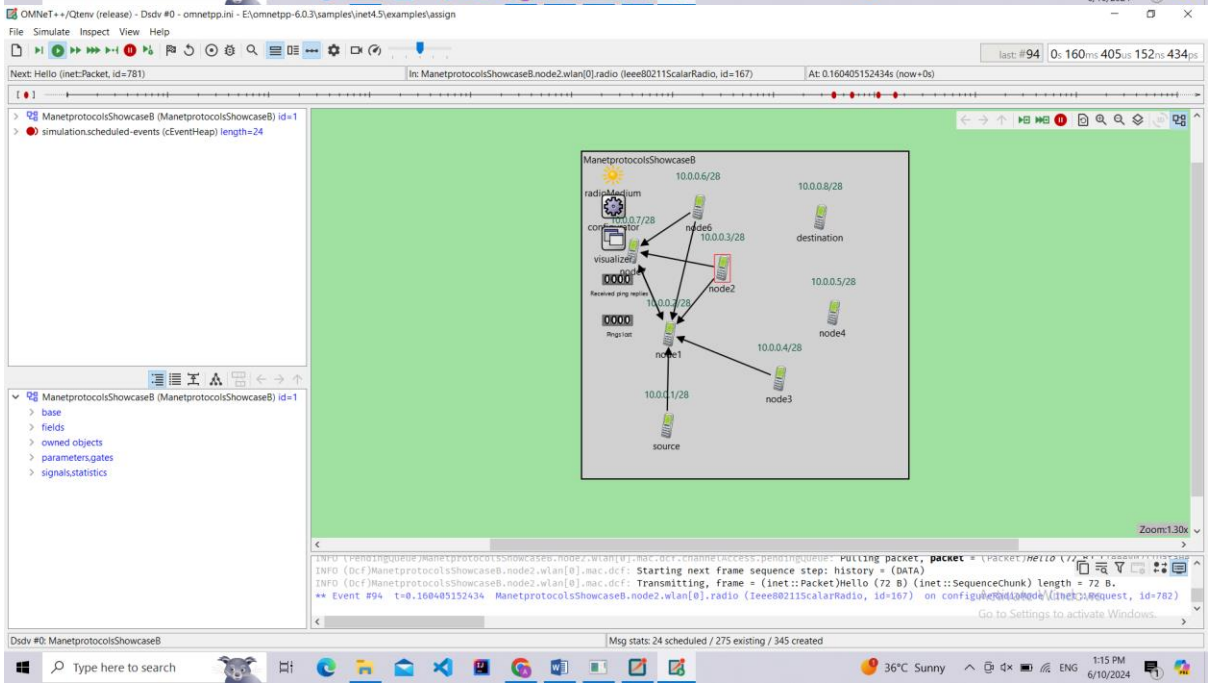

OMNeT++/Qtenv (release) - Aodv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: ping0 (inet::physicallayer::WirelessSignal, id=1508)
In: ManetprotocolsShowcaseB.node2.wlan[0].radio (ieee80211ScalarRadio, id=167)
At: 1.181422752606s (now+0.000000176585s)
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=33
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
radioMedium
10.0.0.6/28
10.0.0.7/28
10.0.0.8/28
node6
destination
visualizer
node7
10.0.0.3/28
10.0.0.2/28
node2
10.0.0.5/28
node1
node4
10.0.0.4/28
node3
source
Zoom:1.30x
** Event #795 t=1.181422576021 ManetprotocolsShowcaseB.node1.wlan[0].radio (ieee80211ScalarRadio, id=102) on ping0 (inet::Packet, id
INFO: Transmission started: (inet::physicallayer::WirelessSignal)ping0 (70 us 135 B) (inet::Packet)ping0 (135 B) (inet::SequenceChunk) length = 135 B WHOLE
INFO: Changing radio transmission state from IDLE to TRANSMITTING.
INFO: Changing radio transmitted signal part from NONE to WHOLE.
Aodv #0: ManetprotocolsShowcaseB
Msg stats: 33 scheduled / 442 existing / 1082 created
Type here to search
Nifty smlcap +1.40%
ENG
1:12 PM
6/10/2024

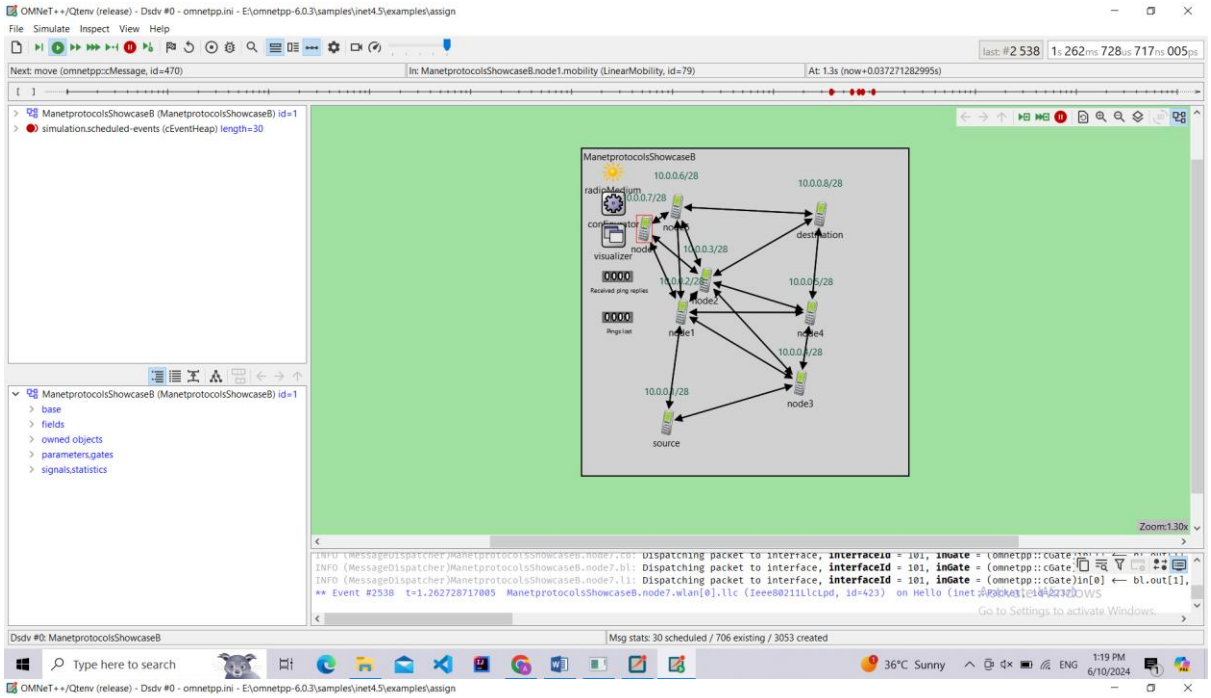

OMNeT++/Qtenv (release) - Aodv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: WlanAck (inet::Packet, id=1633)
In: ManetprotocolsShowcaseB.destination.wlan[0].radio (Ieee80211ScalarRadio, id=492)
At: 1.18206759575s (now+0s)
next: #884
1s 182ms 067us 595ns 750ps
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=31
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
radioMedium
10.0.0.6/28
10.0.0.8/28
10.0.0.7/28
ping0
destination
node6
visualizer
node7
10.0.0.3/28
ping0-reply
10.0.0.2/28
node2
10.0.0.5/28
node1
node4
10.0.0.4/28
10.0.0.1/28
node3
source
Zoom:1.30x
INFO: Radio mode changed from RECEIVER to TRANSMITTER.
INFO: Changing radio reception state from IDLE to UNDEFINED.
INFO: Changing radio transmission state from UNDEFINED to IDLE.
INFO: Changing radio received signal part from WHOLE to NONE.
Activate Windows
Aodv #0: ManetprotocolsShowcaseB
Msg stats: 31 scheduled / 453 existing / 1197 created
Type here to search
Nifty smlcap +1.40%
ENG
1:13 PM
6/10/2024

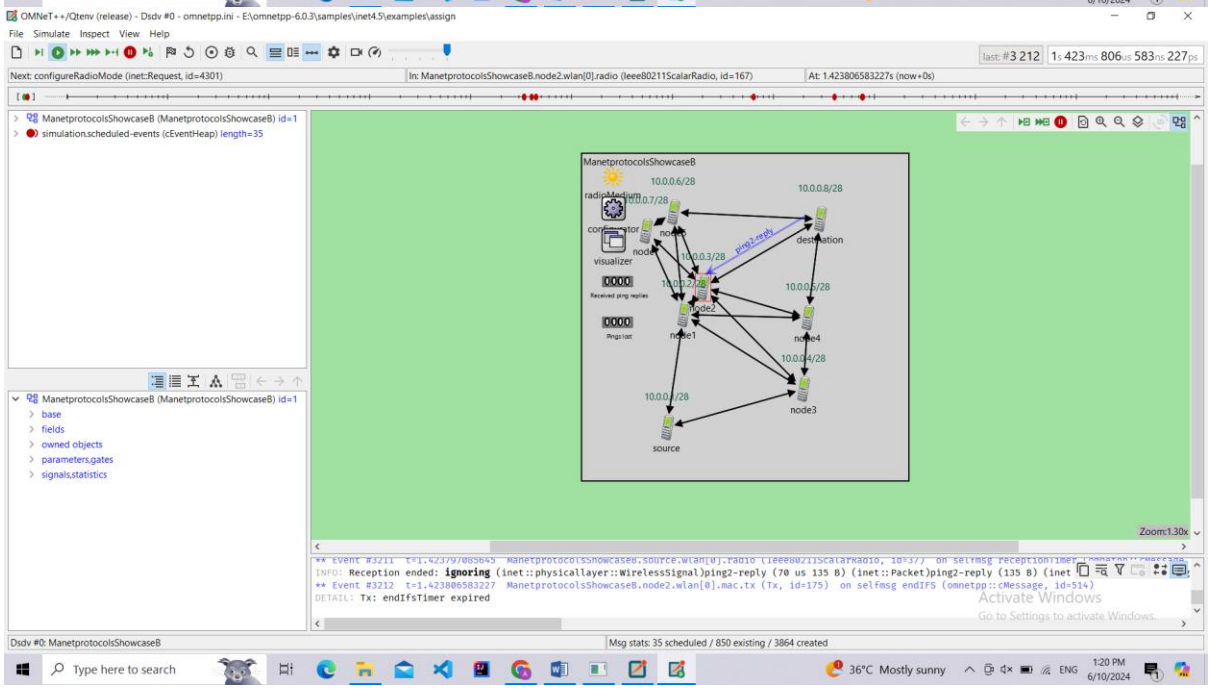

OMNeT++/Qtenv (release) - Aodv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: configureRadioMode (inet::Request, id=1888)
In: ManetprotocolsShowcaseB.source.wlan[0].radio (Ieee80211ScalarRadio, id=37)
At: 1.183681063442s (now+0s)
next: #1 053
1s 183ms 681us 063ns 442ps
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=34
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
radioMedium
10.0.0.6/28
10.0.0.8/28
10.0.0.7/28
destination
node6
visualizer
node7
10.0.0.3/28
ping0-reply
node2
10.0.0.5/28
node1
node4
10.0.0.4/28
ping0-reply
10.0.0.1/28
node3
source
Zoom:1.30x
** Event #1051 t=1.183671315714 ManetprotocolsShowcaseB.destination.wlan[0].radio (Ieee80211ScalarRadio, id=492) on selfmsg receptio
INFO: Reception ended: ignoring (inet::physicallayer::Wireless5ignal)ping0-reply (70 us 135 B) (inet::Packet)ping0-reply (135 B) (inet::SequenceChunk) leng
** Event #1052 t=1.183681063442 ManetprotocolsShowcaseB.source.wlan[0].mac.tx (Tx, id=45) on selfmsg endIFS (omnetpp::cMessage, id=451)
DETAIL: Tx: endIfsTimer expired
Activate Windows
Aodv #0: ManetprotocolsShowcaseB
Msg stats: 34 scheduled / 499 existing / 1451 created
Type here to search
Nifty smlcap +1.40%
ENG
1:13 PM
6/10/2024

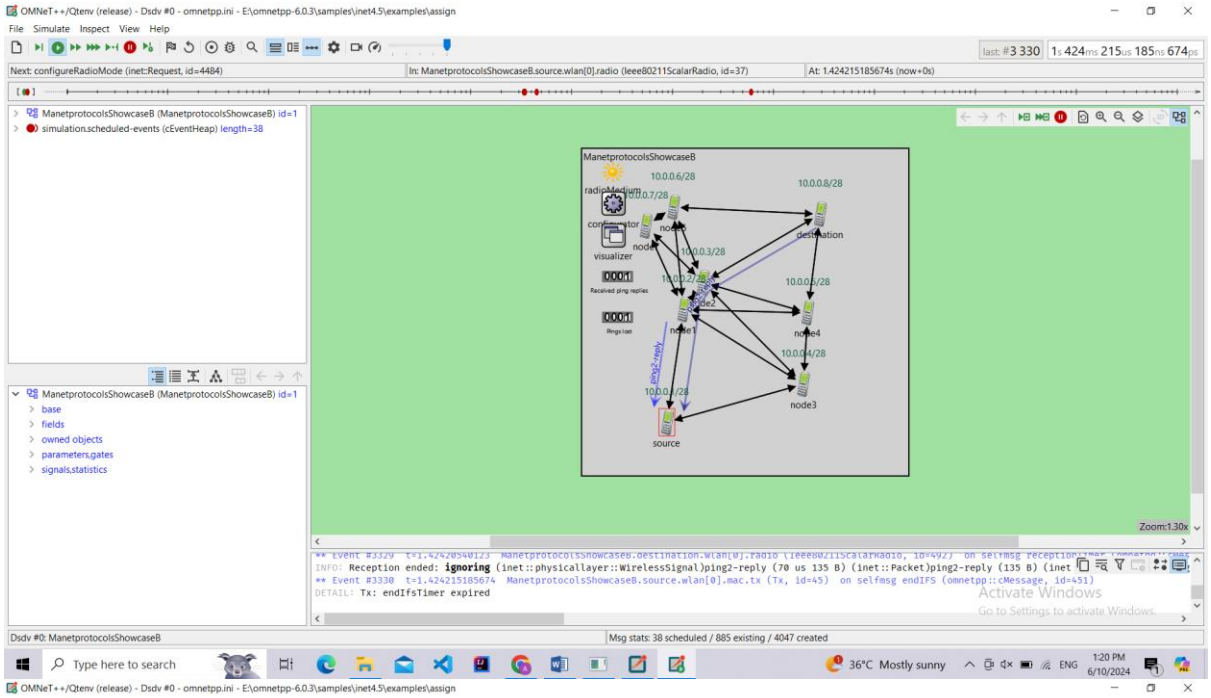
DSDV PROTOCOL

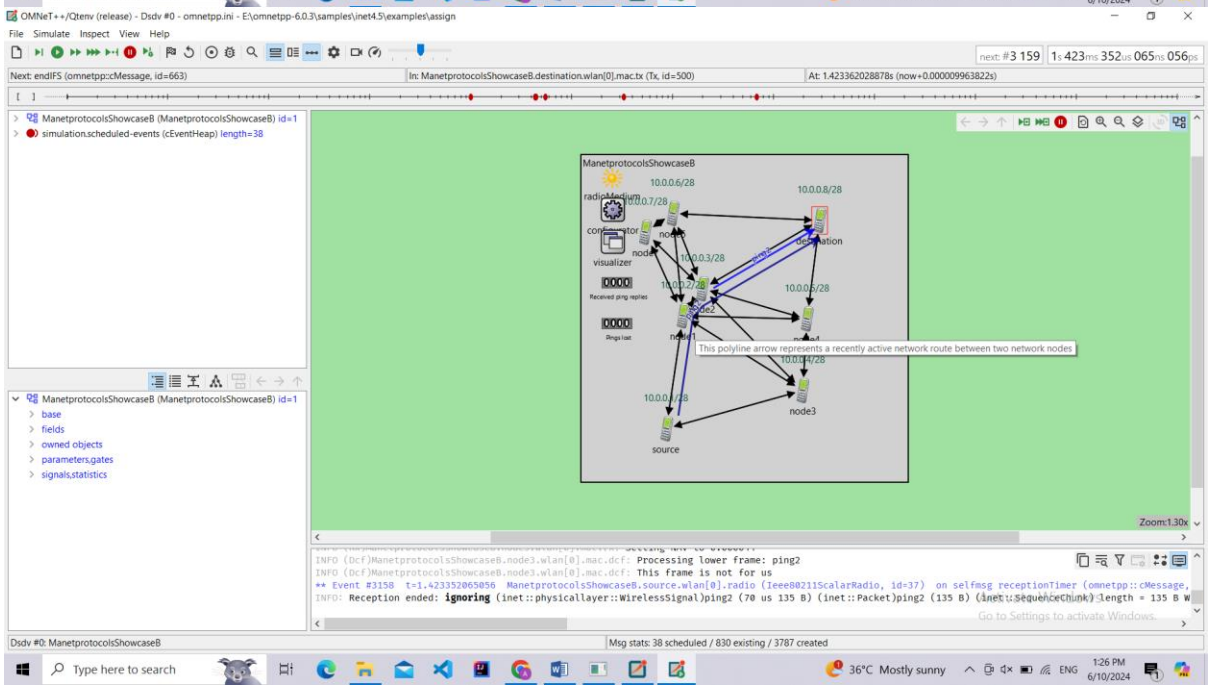





OMNeT++/Qtenv (release) - Dsdv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: move (omnetpp::cMessage, id=470)
In: ManetprotocolsShowcaseB.node1.mobility (LinearMobility, id=79)
At: 1.3s (now+0.037271282995s)
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=30
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
radioMedium
configurator
visualizer
Received ping replies
Pings lost
10.0.0.6/28
10.0.0.7/28
10.0.0.8/28
10.0.0.3/28
10.0.0.2/28
10.0.0.5/28
10.0.0.4/28
10.0.0.1/28
node1
node2
node3
node4
source
destination
Zoom:1.30x
Dsdv #0: ManetprotocolsShowcaseB
Msg stats: 30 scheduled / 706 existing / 3053 created
Type here to search
36°C Sunny
ENG
1:19 PM
6/10/2024


OMNeT++/Qtenv (release) - Dsdv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: configureRadioMode (inet::Request, id=4301)
In: ManetprotocolsShowcaseB.node2.wlan[0].radio (Ieee80211ScalarRadio, id=167)
At: 1.423806583227s (now+0s)
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=35
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
radioMedium
configurator
visualizer
Received ping replies
Pings lost
10.0.0.6/28
10.0.0.7/28
10.0.0.8/28
10.0.0.3/28
10.0.0.2/28
10.0.0.5/28
10.0.0.4/28
10.0.0.1/28
node1
node2
node3
node4
source
destination
ping2-reply
Zoom:1.30x
INFO: Reception ended: ignoring (inet::physicallayer::Wireless5ignal)ping2-reply (70 us 135 B) (inet::Packet)ping2-reply (135 B)
** Event #3212 t=1.423806583227 ManetprotocolsShowcaseB.node2.wlan[0].mac.tx (Tx, id=175) on selfmsg endIFS (omnetpp::cMessage, id=514)
DETAIL: Tx: endIfsTimer expired
Activate Windows
Go to Settings to activate Windows.
Dsdv #0: ManetprotocolsShowcaseB
Msg stats: 35 scheduled / 850 existing / 3864 created
Type here to search
36°C Mostly sunny
ENG
1:20 PM
6/10/2024


OMNeT++/Qtenv (release) - Dsdv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: configureRadioMode (inet::Request, id=4484)
In: ManetprotocolsShowcaseB.source.wlan[0].radio (Ieee80211ScalarRadio, id=37)
At: 1.424215185674s (now+0s)
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=38
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
10.0.0.6/28
10.0.0.8/28
10.0.0.7/28
10.0.0.3/28
10.0.0.2/28
10.0.0.5/28
10.0.0.4/28
10.0.0.1/28
destination
node2
node1
node4
node3
source
visualizer
Received ping replies
Pings lost
ping2-reply
Zoom:1.30x
INFO: Reception ended: ignoring (inet::physicallayer::WirelessSignal)ping2-reply (70 us 135 B) (inet::Packet)ping2-reply (135 B) (inet
** Event #3330 t=1.424215185674 ManetprotocolsShowcaseB.source.wlan[0].mac.tx (Tx, id=45) on selfmsg endIFS (omnetpp::cMessage, id=451)
DETAIL: Tx: endIfsTimer expired
Dsdv #0: ManetprotocolsShowcaseB
Msg stats: 38 scheduled / 885 existing / 4047 created
Type here to search
36°C Mostly sunny
ENG
1:20 PM
6/10/2024


OMNeT++/Qtenv (release) - Dsdv #0 - omnetpp.ini - E:\omnetpp-6.0.3\samples\inet4.5\examples\assign
File Simulate Inspect View Help
Next: endIFS (omnetpp::cMessage, id=663)
In: ManetprotocolsShowcaseB.destination.wlan[0].mac.tx (Tx, id=500)
At: 1.423362028878s (now+0.000009963822s)
ManetprotocolsShowcaseB (ManetprotocolsShowcaseB) id=1
simulation.scheduled-events (cEventHeap) length=38
base
fields
owned objects
parameters,gates
signals,statistics
ManetprotocolsShowcaseB
10.0.0.6/28
10.0.0.8/28
10.0.0.7/28
10.0.0.3/28
10.0.0.2/28
10.0.0.5/28
10.0.0.4/28
10.0.0.1/28
destination
node2
node1
node4
node3
source
visualizer
Received ping replies
Pings lost
This polyline arrow represents a recently active network route between two network nodes
Zoom:1.30x
INFO (Dcf)ManetprotocolsShowcaseB.node3.wlan[0].mac.dcf: Processing lower frame: ping2
INFO (Dcf)ManetprotocolsShowcaseB.node3.wlan[0].mac.dcf: This frame is not for us
** Event #3158 t=1.423352065056 ManetprotocolsShowcaseB.source.wlan[0].radio (Ieee80211ScalarRadio, id=37) on selfmsg receptionTimer (omnetpp::cMessage,
INFO: Reception ended: ignoring (inet::physicallayer::WirelessSignal)ping2 (70 us 135 B) (inet::Packet)ping2 (135 B) (inet::SequenceChunk) length = 135 B W
Dsdv #0: ManetprotocolsShowcaseB
Msg stats: 38 scheduled / 830 existing / 3787 created
Type here to search
36°C Mostly sunny
ENG
1:26 PM
6/10/2024