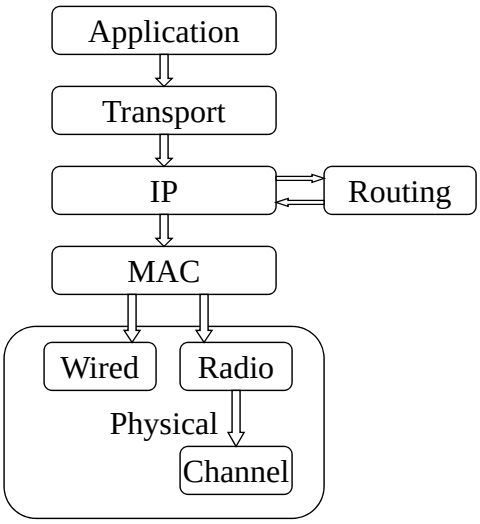
Using QualNet – Part I

Simulating Well-Known Protocols

Why QualNet?

- Rapid prototyping of protocols
- Comparative performance evaluation of alternative protocols at each layer
- Built-in measurements on each layer
- Modular, layered stack design
- Standard API for composition of protocols across different layers
- Scalability via support for parallel execution & dual-core processor
- GUI Tools for system/protocol modeling

QualNet Layer Model



FTP, Telnet, CBR

TCP, UDP

OSPF, AODV

802.3, 802.11, CSMA

Radio, Bus, Point to Point

Free space, Two-ray

Installing QualNet

- Refer to *QualNet-4.0-InstallationGuide.pdf*
- Obtain QualNet 4.0 package from *stimpy.cis*

```
$ cd <working_dir>
$ gtar xvfz
/degas/research/simulators/qualnet/4.0/
  qualnet-4.0-university-wireless-mme.tar.gz
```

- Obtain the license file

```
$ cd qualnet/4.0
$ cp
/degas/research/simulators/qualnet/license/client.lic
license_dir/.
```

Note: for successful license check, QualNet ***must*** be installed on machines within EECIS domain (128.4.*.*), ***not*** even UD domain

Installing QualNet (cont'd)

- Set QUALNET_HOME/JDK_HOME environment variable

```
$ setenv QUALNET_HOME <working_dir>/qualnet/4.0
$ QUALNET_HOME=<working_dir>/qualnet/4.0
$ export QUALNET_HOME
```

csh/tcsh

sh/bash

- Build an executable

```
$ cd main
$ cp Makefile-solaris Makefile
$ make
```

Platform-dependent (cpu/os/compiler);
Note:(1) solaris/sparc supported, but *not* solaris/x86; (2) redefine "CC/CXX" in *Makefile-unix/windows-common* for a different gcc version if needed

- Test QualNet's installation

```
$ cd ../bin
$ ./qualnet default.config
```

```
$ cd ../gui/netbeans/bin
$ runide.sh
```


Running QualNet

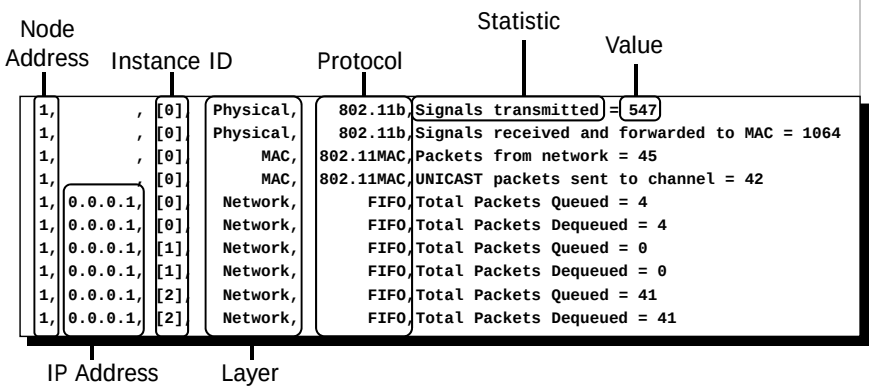
Default configuration file

```
$ ./qualnet default.config
QUALNET_HOME = /usa/kli/qualnet/4.0
QualNet Version 4.0
Attempting license checkout (should take less than 2 seconds) ... success.
Partition 0, Node 1 (210.35, 159.63, 0.00).
Partition 0, Node 2 (479.92, 134.36, 0.00).
:
Partition 0, Node 30 (1391.27, 1219.75, 0.00).

Initialization completed in   0.2608 sec

Current Sim Time[s] =    0.000000000  Real Time[s] =    0  Completed  0%
Current Sim Time[s] =    9.046229850  Real Time[s] =    0  Completed  1%
:
Current Sim Time[s] =  891.040233635  Real Time[s] =    1  Completed 99%
Current Sim Time[s] =  900.000000000  Real Time[s] =    1  Completed100%
Executed 379907 events in   2.3882 sec

$
```

Initial node's coordinate (x,y,z)

Simulation Progress

Output Statistics – `default.stat`



* *perl* and *awk* are great tools for processing statistic outputs

Configuration Files

- Line entry format:

  `[Qualifier] <PARAMETER> <VALUE>`

  - Qualifier (optional) specifies a range of nodes and has precedence over the general one
  - E.g.

```
MOBILITY NONE
[5 thru 10] MOBILITY RANDOM-WAYPOINT
```

- Notes:
  - Some settings require additional parameters, e.g. NODE-PLACEMENT
  - Lines starting with # are treated as comments
  - Refer to scenarios/default/default.config

Qualifiers

- Determine the scope of the parameter
  - Global Qualifier
    ```
    MOBILITY  NONE
    ```
  - Subnet Qualifier
    ```
    [N8-2.0] MAC-PROTOCOL  MACA
    ```
  - Node Qualifier
    ```
    [5 thru 15] MOBILITY NONE
    ```


Simulation Parameters

- Global Parameters:
  - Simulation time
  - Coordinate system and terrain
  - Random seed
- Topology and subnets
- Layer/Protocol related parameters:
  - Channel/Radio
  - Physical Layer
  - MAC Layer
  - Network Layer
  - Application Layer

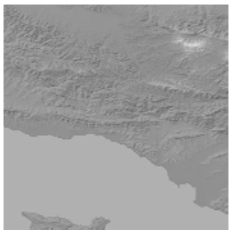
Important Global Parameters

- EXPERIMENT-NAME: Name of the output statistic file
  - e.g. **`EXPERIMENT-NAME default`**
  - Resulting statistics are written in **`default.stat`**
- SIMULATION-TIME: The length of time to simulate.
  - e.g. **`SIMULATION-TIME 15M`**
  - (Available time units: NS, US, MS, S, M, H, D; default is in seconds)
- SEED: The random seed used to derive all other seeds used in the simulation.
  - e.g. **`SEED 1`**

Coordinate System and Terrain Dimentions

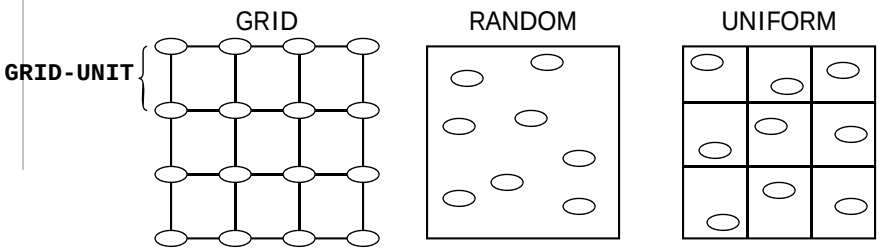
- COORDINATE-SYSTEM: LATLONALT or CARTESIAN
  - e.g., **`COORDINATE-SYSTEM CARTESIAN`**
- TERRAIN-DIMENSIONS: The size of the rectangular area to simulate (in meters) for Cartesian coordinate.
  - e.g., **`TERRAIN-DIMENSIONS (1000, 1000)`**
- Terrain corners are required by LATLONALT system
  - e.g., **`TERRAIN-SOUTH-WEST-CORNER (30.00, 40.00)`**
    `TERRAIN-NORTH-EAST-CORNER (30.01, 40.01)`
- Irregular terrain

```
TERRAIN-DATA-TYPE DEM
DEM-FILENAME[0] ../data/terrain/los_angeles-w
DEM-FILENAME[1] ../data/terrain/los_angeles-e

# Tie nodes to the ground level
MOBILITY-GROUND-NODE YES
```



Specifying Topology

- NODE-PLACEMENT: GRID, RANDOM, UNIFORM, FILE



- Use FILE to specify node positions in a file

```
NODE-PLACEMENT      FILE
NODE-PLACEMENT-FILE ./default.nodes
```

  - Format: *nodeId 0 (x, y, z) [azimuth elevation]*

Specifying Subnet

- SUBNET Parameter
  - SUBNET <subnet> { comma-delimited list of nodes }
- Ex. **`SUBNET N8-1.0 { 1, 3, 7 thru 9 }`**
  - NodeIds 1, 3, 7, 8, and 9 have network interfaces with address 0.0.1.1 through 0.0.1.5

Node ID	Interface Address
1	0.0.1.1
3	0.0.1.2
7	0.0.1.3
8	0.0.1.4
9	0.0.1.5

- Basic form: **`SUBNET N16-0 { 1 thru `*`n`*` }`**
  - *n* is the number of nodes
  - IP address and Node ID are identical
    → Node 5 has IP address 0.0.0.5

Subnet Shorthand

- Format:

  N<# host bits>**-**<address with front end 0's omitted>

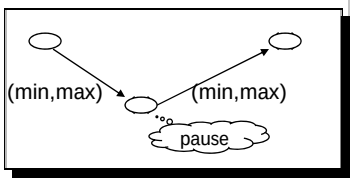
- Ex. *N8-1.0*
  - The subnet address is 0.0.1.0
  - Host IP addresses are 8 bits

  - This allows for 2^8-2 (254) hosts in this subnet with IP addresses numbered from 0.0.1.1 through 0.0.1.254
  - The broadcast address for this subnet is 0.0.1.255
  - The subnet mask is 255.255.255.0
  - *N8-0.0.1.0* is an equivalent representation

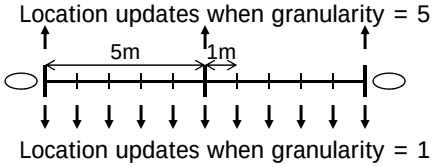
Mobility Model

- NONE, TRACE
- RANDOM-WAYPOINT

  e.g. **MOBILITY RANDOM-WAYPOINT**
  MOBILITY-WP-PAUSE 30S
  MOBILITY-WP-MIN-SPEED 0
  MOBILITY-WP-MAX-SPEED 10



- MOBILITY-POSITION-GRANULARITY: distance in meters at which a node's location is updated
  - ☞ small values potentially slow down the simulation

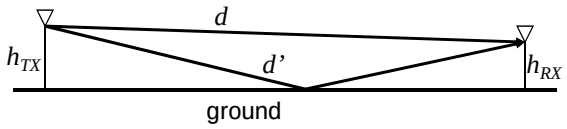


Approximate mobility speed

Scenario	Speed	Speed (m/s)
Walking	5 mph	2.2 m/s
City driving	35 mph	15.5 m/s
Free way driving	65 mph	28.8 m/s
Aircraft	Mach 1	332 m/s

Propagation Model

- PROPAGATION-LIMIT: received signals with power below this limit (in dBm) will not be processed.
  e.g. **`PROPAGATION-LIMIT  -111.0`**
- PROPAGATION-PATHLOSS: specifies path-loss model
  - FREE-SPACE → Empty space, no ground
  - TWO-RAY → Flat ground
    - Considers a ray bounced back from the ground



  - ITM → Irregular terrain (terrain database required)

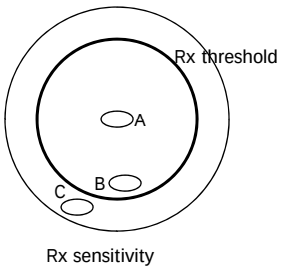
Fading Model

- Applied to only narrowband channels (flat fading)
- Specified by PROPAGATION-FADING-MODEL
- Available models
  - NONE – No fading
  - RAYLEIGH – Highly mobile, no line of sight
  - RICEAN – requires an additional parameter RICEAN-K-FACTOR
    - K = 0 : no line of sight (similar to RAYLEIGH)
    - K = ∞ : strong line of sight

e.g. **`PROPAGATION-FADING-MODEL RAYLEIGH`**

Physical Layer Model

- Noise modeling
  - Thermal noise
    - PHY-NOISE-FACTOR (default 10)
    - PHY-TEMPERATURE (in K; default 290)
  - Interference
    - PHY-RX-MODEL (SNR-THRESHOLD-BASED | BER-BASED | 802.11b)
- Parameters specific to 802.11
  - PHY802.11-DATA-RATE (in bps)
  - PHY802.11b-TX-POWER-* (in dBm)
  - PHY802.11b-RX-SENSITIVITY-* (in dBm)
  - PHY802.11b-RX-THRESHOLD-* (in dBm)



MAC Layer Model

- MAC-PROTOCOL: specifies MAC layer protocol
  - CSMA
    - Requires carrier sensing before transmission
    - If the channel is free, the packet is transmitted immediately
    - Otherwise, set a random timeout
  - MACA
    - Uses RTS/CTS to acquire channel
    - Does not carrier sense
  - MACDOT11
    - IEEE 802.11 CSMA/CA with ACKs and optional RTS/CTS
  - MAC802.16 → WiMAX
  - TDMA, GSM, ALOHA
  - MAC802.3/SWITCHED-ETHERNET → Wired networks
  - SATCOM → Satellite networks
- PROMISCUOUS-MODE: set to YES to allow nodes to overhear packets destined to the neighboring node (required by DSR).

IP Protocol

- Currently the only supported network layer protocol
- Available queuing models
  - First-in first-out (FIFO)
  - Variations of Random Early Detection: RED, RIO, WRED,
- Three priority types supported: *control* (0), *real-time* (1), and *non-real-time* (2)
- IP-QUEUE-PRIORITY-QUEUE-SIZE specifies the queue's size (in bytes)
  - Each priority queue's size can be specified separately

```
IP-QUEUE-PRIORITY-QUEUE-SIZE[0] 25000
IP-QUEUE-PRIORITY-QUEUE-SIZE[1] 50000
IP-QUEUE-PRIORITY-QUEUE-SIZE[2] 50000
```

Routing Protocols

- Proactive protocols
  - BELLMANFORD
  - RIPv2
  - OSPFv2
  - OLSR-INRIA
- Reactive protocols
  - AODV
  - DSR
  - LAR1
- Static routing: requires STATIC-ROUTE-FILE

Application Specification

- APP-CONFIG-FILE: Specifies a file with a list of apps/traffic generators to run.
  - FTP
  - TELNET
  - CBR/MCBR
  - HTTP
  - VOIP
  - *etc*
- See scenarios/default/default.app for more details

Layer Statistics

- Some statistics include:
  - APPLICATION-STATISTICS (YES|NO)
  - TCP-STATISTICS (YES|NO)
  - UDP-STATISTICS (YES|NO)
  - ROUTING-STATISTICS (YES|NO)
  - NETWORK-LAYER-STATISTICS (YES|NO)
  - QUEUE-STATISTICS (YES|NO)
  - MAC-LAYER-STATISTICS (YES|NO)
  - PHY-LAYER-STATISTICS (YES|NO)
  - MOBILITY-STATISTICS (YES|NO)

Other Configuration Files

- Node placement: **NODE-PLACEMENT-FILE**
  - → See details in **scenarios/default/default.nodes**
- Static routing: **STATIC-ROUTE-FILE**
  - → See details in **scenarios/default/default.routes-static**
- Link/node faults: **FAULT-CONFIG-FILE**
  - → See details in **scenarios/default/default.fault**
- Multicast membership: **MULTICAST-GROUP-FILE**
  - → See details in **scenarios/default/default.member**

Example – routing protocol comparison

- Comparing throughput of an FTP application over AODV and DSR reactive routing protocols
  - Application traffic: one FTP session
  - Routing protocols: AODV/DSR
  - Collected statistics: application layer
- Configure application
  - Create **`bin/ftp.app`** containing only one line:

```
FTP 1 30 10 150S
```

  - The session starts at 150 secs.
  - There are 10 items to be sent
  - Node 30 is the server (receiver)
  - Node 1 is the client (sender)

Preparing Config File for AODV

- *AODV:* copy the file **`default.config`** to **`aodv.config`**, then modify **`aodv.config`** on the following parameters:

```
EXPERIMENT-NAME                          aodv
ROUTING-PROTOCOL                         AODV
APP-CONFIG-FILE                          ./ftp.app
APPLICATION-STATISTICS                   YES
TCP-STATISTICS                           NO
UDP-STATISTICS                           NO
RSVP-STATISTICS                          NO
ROUTING-STATISTICS                       NO
ACCESS-LIST-STATISTICS                   NO
IGMP-STATISTICS                          NO
EXTERIOR-GATEWAY-PROTOCOL-STATISTICS     NO
NETWORK-LAYER-STATISTICS                 NO
DIFFSERV-EDGE-ROUTER-STATISTICS          NO
QUEUE-STATISTICS                         NO
MAC-LAYER-STATISTICS                     NO
PHY-LAYER-STATISTICS                     NO
MOBILITY-STATISTICS                      NO
```

Preparing Config File for DSR

- *DSR:* Copy the modified **aodv.config** to **dsr.config**, edit ROUTING-PROTOCOL parameter and enable promiscuous mode:

```
EXPERIMENT-NAME     dsr
ROUTING-PROTOCOL    DSR
PROMISCUOUS-MODE    YES
```


Collecting and Comparing Statistics

- Run qualNet on the two configuration files

```
$ cd $QUALNET_HOME/bin
$ ./qualnet aodv.config
$ ./qualnet dsr.config
```

- Examine output statistics:

Throughput difference

```
$ grep "Server,Throughput" aodv.stat
  30, , [2], Application,  FTP Server,Throughput (bits/s) = 38034
$ grep "Server,Throughput" dsr.stat
  30, , [2], Application,  FTP Server,Throughput (bits/s) = 62946
```

More Information

- Plain-text explanation for configuration files:
  - **`$QUALNET_HOME/scenarios/default/default.*`**
- QualNet manuals
  - **`$QUALNET_HOME/documentation/*.pdf`**

    `esp. Installation Guide, Users Guide, and Programmers Guide`
- QualNet community forums
  - http://www.scalable-networks.com/forums/
- QualNet exercises
  - http://degas.cis.udel.edu/QualNet/