Whitefield Documentation

Rahul Jadhav

Oct 04, 2021

Contents:

CHAPTER 1

About

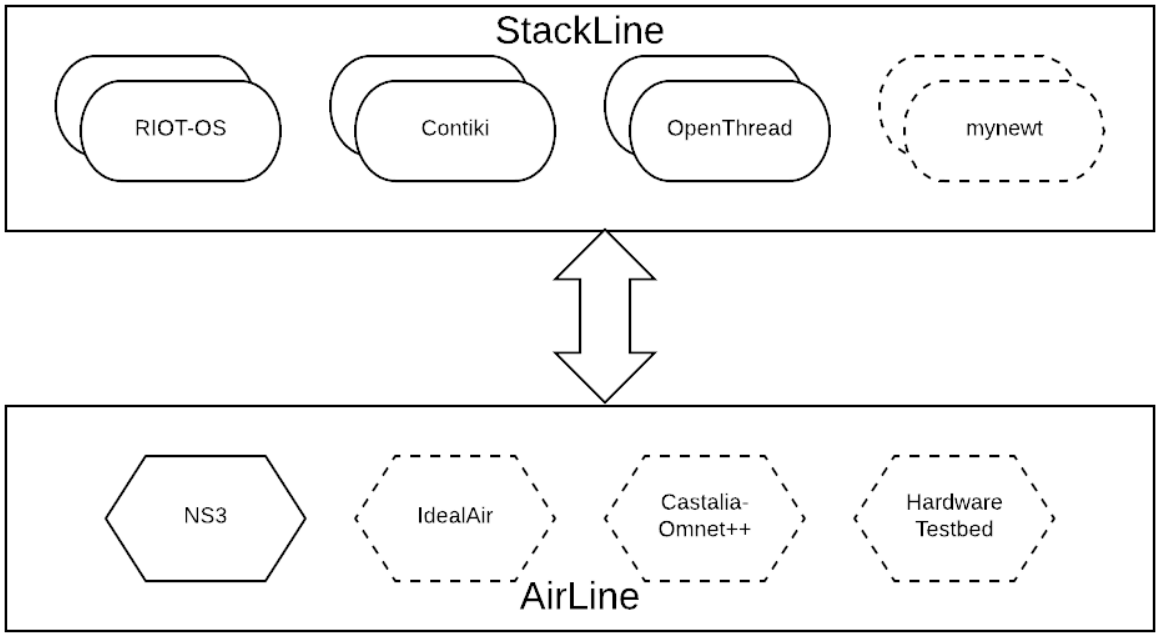
Whitefield provides a simulation environment for wireless sensor network by combining realistic RF simulation with the native mode use of popular IoT stacks/OSes such as Contiki/RIOT/OpenThread. Thus one can use existing stack implementation as is and test it on top of realistic RF simulation.

Typical use of Whitefield:

1. Large scale testing of sensor networks
2. Large scale interop testing between multitudes of stacks in same wireless environment. It can scale to thousands of nodes on general purpose server (even laptops).
3. Validate RF phy/mac simulation against each other. For e.g. compare results in NS3 vs Castalia
4. Regression tests using IdealAir(future).

CHAPTER 2

Terminology



Whitefield decouples the physical layer from above layers.

2.1 Airline

Airline refers to the physical(+mac) layer. Airline implementation is handled by thirdparty simulators such as NS3.

2.2 Stackline

These are the real-world protocol stacks. 6lowpan/Network layer and above protocol stack provided by thirdparty IoT OSes such as Contiki/RIOT. A new platform is added in such OSes to interface with Whitefield. The protocol stack with application will be used as-is from these OSes.

2.3 Commline

Message queue and corresponding interfaces which decouples Airline and Stackline. All the messages be it the network payloads or OAM messages are transported using Commline.

2.4 NodeID

Whitefield assigns an internal nodeid to all the nodes. Note that this node id may not necessarily translate to the mac-address. The nodeid starts with 0 and reaches numOfNodes-1, where numOfNodes is part of the configuration. The wfshell commands refers to this NodeID when invoking a cmd_* command.

CHAPTER 3

Installation

3.1 One Time Setup

```
git clone https://github.com/whitefield-framework/whitefield
cd whitefield
./scripts/setup.sh #This may take a while...
```

setup.sh does following:

1. Checks and installs dependencies
2. Enables specific stacklines and NS3 configuration. This step creates *config.inc*.
3. *make* all the enabled modules

CHAPTER 4

Getting Started

4.1 Updating Configuration

The first step to do post setup is to update the configuration. The *config* folder contains all the configuration files. Check sample configuration wf.cfg. Check detailed configuration options.

4.2 Starting Whitefield

```
cd whitefield
./invoke_whitefield config/wf.cfg
```

4.3 Stopping Whitefield

```
./scripts/wfshell stop_whitefield
```

4.4 wfshell

```
 ./scripts/wfshell
wfsh# <tab><tab>
cmd_config_info         cmd_ipv6_stats          cmd_node_osname         cmd_rtsize
↪       cmd_udp_stats           path_downstream
cmd_def_route           cmd_mac_stats           cmd_node_position       cmd_set_node_
↪position  exit                     path_upstream
cmd_get_udpapp_stat     cmd_nd6_stats           cmd_route_table         cmd_start_udp
↪       help                    plot_network_graph
cmd_icmp_stats          cmd_node_exec           cmd_rpl_stats           cmd_tcp_stats
↪       native_shell            stop_whitefield
```

(continues on next page)

(continued from previous page)

wfshell sends the command to Airline or Stackline and prints the response. The command can be sent to all nodes or to specific nodes.

```
./scripts/wfshell <cmd_foobar> ... executes cmd_foobar on all the nodes
./scripts/wfshell <cmd_foobar> <node_id> ... executes cmd_foobar on node_id only
```

CHAPTER 5

Configuration

Whitefield configuration provides common configuration options across different airlines and stacklines used i.e. the same configuration works whether you use NS3 or Castalia-Omnet++ as Airline or use Contiki/RIOT as stackline.

5.1 Common Config

Key	Value Range	Remarks
numOfNodes	[2-5000]	Number of nodes in the network
fieldX	Uint range	Field length in X direction … Currently only 2D model is supp
fieldY	Uint range	Field length in Y direction
topologyType	randrect	Randomly place nodes in area denoted by fieldX * fieldY
	grid	Grid topology where nodes are separated by distance specified by fieldX * fieldY and the width of the grid is specified by gridWidth
gridWidth	Uint range	Width of the grid. Only applicable when topologyType=grid
txPower[*]	double	Transmit Power in unit dBm
nodePosition[*]	10,20,0	Manually position the node at the given location specified by x,y,z coordinates
nodePromiscuous[*]	1	Set promiscuous mode for the node. Node will receive not only broadcast or unicast packets destined to it but also other packets not destined to it but the node is in receive range.
panID	Ushort range	PAN identifier to be used in LOWPAN
macPktQlen	<100	Maximum number of packets that can be buffered/queued at MAC layer
macMaxRetry	<20	Maximum number of times the mac packet will be retried
nodeExec[*]	/path/to/stack/binar	Native compiled executable path for Contiki/RIOT nodes will be specified here
captureFile[*]	/path/to/pcap_dir	Location where pcap will be stored … Not supported currently, use NS3_captureFile instead
NS3_captureFile	/path/to/pcap_dir	Uses NS3's inbuilt pcap capturing method
include	/path/to/include_file	Include configuration from other file
PHY	[plc,lr-wpan]	Default lr-wpan if this is not specified
lossModel	*supported loss models*	Default none
lossModelParam	key=value pairs	Dependent on corresponding lossModel
delayModel	*supported delay models*	Default none
delayModelParam	key=value pairs	Dependent on corresponding delayModel

The configuration can be applied to only a set of nodes (for configuration options specified with [*]) by specifying the node index range (note, the first node has an index of zero). For e.g.

```
numOfNodes=20
nodeExec=/path/to/contiki
```

(continues on next page)

(continued from previous page)

```
nodeExec[5]=/path/to/scapy
nodeExec[10-19]=/path/to/riot
```

In the above configuration the first nodeExec=/path/to/contiki will result in the execuatable getting set for all nodes. In the subsequent config statement, nodes[10-19] (inclusive) will override the default nodeExec path.

Note that the sequence of configuration option is important.

5.2 Propagation Loss Models

5.2.1 LogDistance (def)

NS3 help

Key	Value Range	Remarks
PathLossExp	double	Check NS3 help
RefDist	double	Check NS3 help
RefLoss	double	Check NS3 help

Note: RefDist and RefLoss has to be specified together, otherwise individually it is ignored.

5.2.2 Friis

NS3 help

Key	Value Range	Remarks
Freq	double	Frequency: Check NS3 help
minLoss	double	MinLoss: Check NS3 help
sysLoss	double	SystemLoss: Check NS3 help

5.2.3 FixedRss

NS3 help

Key	Value Range	Remarks
rss	double	Check NS3 help

5.2.4 Matrix

NS3 help

Key	Value Range	Remarks
defLoss	double	Check NS3 help

5.2.5 Random

NS3 help

No additional parameters supported.

5.2.6 Range

NS3 help

No additional parameters supported.

5.2.7 ThreeLogDistance

NS3 help

No additional parameters supported.

5.2.8 TwoRayGround

NS3 help

Key	Value Range	Remarks
sysLoss	double	Check NS3 help
Freq	double	Frequency: Check NS3 help
HeightAboveZ	double	Check NS3 help
minDist	double	minDistance: Check NS3 help

5.3 Propagation Delay Models

5.3.1 ConstantSpeed (def)

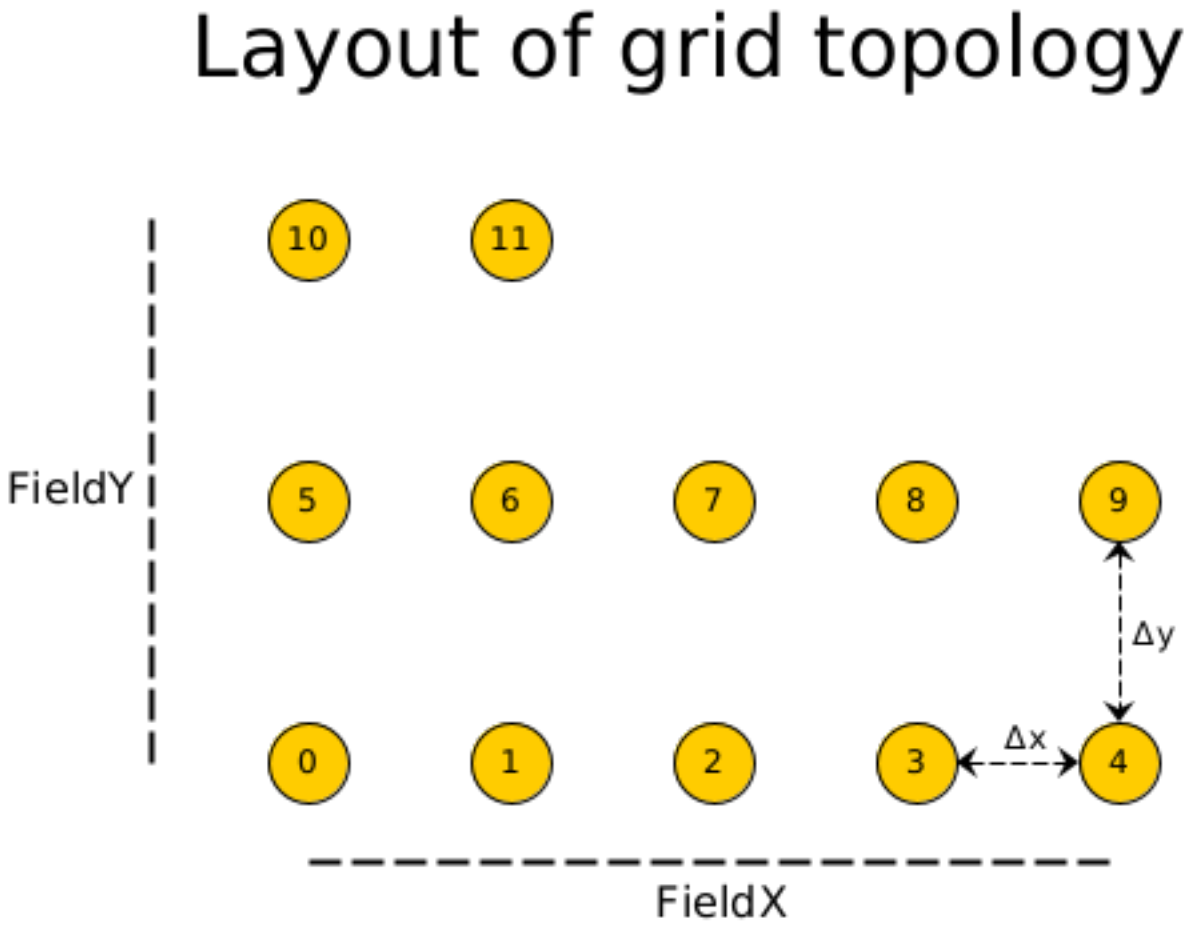
NS3 Help

Key	Value Range	Remarks
speed	double	Check NS3 help

5.3.2 Random

NS3 Help No additional parameters supported.

5.4 Sample layout for Grid topology



Δx = fieldX / gridWidth
Δy = fieldY / (number of rows)
Number of rows = ceil(gridWidth/numOfNodes)

Example config:
---
numOfNodes = 12
fieldX = 200 #meters
fieldY = 200 #meters
topologyType = grid
gridWidth = 5

Thus,
Δx = 200/5 = 40m
Number of rows =ceil(12/5) = 3
Δy = 200/3 = 66m

CHAPTER 6

Module support

The best place to check whats up in the future is Whitefield Project on Trello

6.1 Support Table

Stackline Modules	Supported	Remarks
Contiki	Yes	
Contiki-NG	Yes	
RIOT	Yes	
OpenThread	Future	Part of the support is added but not complete
Zephyr	Future	

Airline Modules	Supported	Remarks
802.15.4	Yes	supports NS3 lrwpan module
PLC	Yes	supports NS3 PLC module
Static Visualization	Yes	Using Graphviz to plot network diagram
Dynamic Visualization	Yes	Canvas tool which plots nodes and their edges dynamically using Cytoscape.js
Castalia-Omnet	Future	Not sure if Omnet license allows integrating with Whitefield
802.15.4e	Future	Not planned yet. Requires submitting changes to NS3
LoRAWAN	Future	Possibly based on NS3-lorawan
WAVE/DSRC	Future	
Runtime Mobility	Future	Currently it is possible to change the node's location using *cmd_set_node_position*

General Modules	Supported	Remarks
Static Visualization	Yes	Using Graphviz to plot network diagram
Dynamic Visualization	Yes	Canvas tool which plots nodes and their edges dynamically using Cytoscape.js

6.2 Advanced Customisations

6.2.1 How to add a new RF simulator using Airline/Commline interfaces

The Airline module contains an RF simulator (such as NS3) which provides PHY and MAC layer. The MAC layer is tightly coupled with the physical layer and hence it is expected that the Airline module (be it NS3/Omnet/Opnet) provides both PHY/MAC layer. One example reason for such tight coupling is the case where the MAC layer depends upon "carrier sensing" interfaces provided by physical layer. Loose coupling these calls may cause severe performance impact since these interfaces are called much too often. Thus the general expectation of the Airline module is: 1. Provide a PHY layer with the desired properties 2. Provide a MAC/LLC layer including MAC framing capability. 3. Receive messages from commline, the message contains the packet buffer (IP layer and above) along with metadata in the form of src node_id, dst node_id. The Airline module is responsible for handing out the packet to the appropriate node's network interface after doing the MAC layer framing. Usually there are interfaces provided by NS3/Omnet/... for handling such tasks and one can look at airline/NS3 implementation to understand better.

Current Limitations: 1. The implementation uses uint16_t as node id. Thus maximum of 65K nodes are allowed in one network. Please note that node_id is used to map to the MAC interface identifier (IID) but the length of MAC IID may be different than uint16_t. For example, the NS3 lr-wpan (RF simulator based on 802.15.4) based Airline module can support short address(2B) or extended address (8B) addressing mode as per design. But the actual implementation of NS3 currently supports only 2B addressing mode so the MAC framing is done using short address mode only.

6.2.2 How to add a new IoT protocol stack using Stackline/Commline interfaces

1. Explain changes to Contiki
2. Explain changes to RIOT

CHAPTER 7

Indices and tables

- genindex
- modindex
- search