DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

1222·2022 800 ANNI

UNIVERSITÀ DEGLI STUDI DI PADOVA

SIGNET
signet.dei.unipd.it

Final exercise

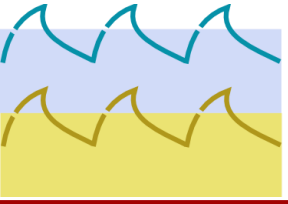
Evo Logics®

UNWiS - Padova (Italy)

30th of January – 3rd of February 2023

Filippo Campagnaro, Roberto Francescon, Emanuele Coccolo, Angela Soldà, Antonio Montanari, Michele Zorzi

filippo.campagnaro@unipd.it



Final exercise

Script running EvoLogics nodes in a sea trial

Emulation script

- The script will be making use of the real time scheduler to be able to connect to actual devices
- It will interface with the EvoLogics modems (in an emulated environment: DMACE®)
- You will make use of the `uwevologicss2cmodem` module
- You will choose the protocols of the stack, and which metrics to measure

Steps

It is highly suggested to follow these steps:

- Set up the script by importing the modules from another exercise, so you won't be needing to set correct order again
- Test, from time to time, when you add new features or protocol entities

Steps of the development

Development of the exercise

Importing the libraries

From the script of the TDMA simulation we can retrieve a base for the imported libraries

```
load libMiracle.so
load libuwip.so
load libuwstaticrouting.so
load libmphy.so
load libmmac.so
load libuwmmac_clmsgs.so
load libuwphy_clmsgs.so
load libuwmll.so
load libuwudp.so
load libuwcbr.so
load libuwtdma.so
load libuwinterference.so
load libUwmStd.so
load libuwphy_clmsgs.so
load libuwstats_utilities.so
load libuwphysical.so
```

Set the simulator engine

Instantiate a new simulator engine and tell it to use Miracle

Then, tell it to use the real time scheduler

```
set ns [new Simulator]
$ns use-Miracle

$ns use-scheduler RealTime
```

Start configuring modules

e.g., the AL layer module

```
Module/UW/AL set Dbit                  0
Module/UW/AL set PSDU                  1400
Module/UW/AL set debug_                0
Module/UW/AL set interframe_period     0.e1
Module/UW/AL set frame_set_validity 0
```

e.g., the Application packer module

```
UW/APP/uwApplication/Packer set SN_FIELD_ 8
UW/APP/uwApplication/Packer set RFFT_FIELD_ 5
UW/APP/uwApplication/Packer set RFFTVALID_FIELD_ 2
UW/APP/uwApplication/Packer set PRIORITY_FIELD_ 8
UW/APP/uwApplication/Packer set PAYLOADMSG_FIELD_SIZE_ 8
UW/APP/uwApplication/Packer set debug_ 10
```

Continue configuring modules

Configure all the needed modules

```
NS2/COMMON/Packer set PTYPE_Bits 8
NS2/COMMON/Packer set SIZE_Bits 8
NS2/COMMON/Packer set UID_Bits 8
NS2/COMMON/Packer set ERROR_Bits 0
NS2/COMMON/Packer set TIMESTAMP_Bits 8
NS2/COMMON/Packer set PREV_HOP_Bits 8
NS2/COMMON/Packer set NEXT_HOP_Bits 38
NS2/COMMON/Packer set ADRR_TYPE_Bits 0
NS2/COMMON/Packer set LAST_HOP_Bits 0
NS2/COMMON/Packer set TXTIME_Bits 0
NS2/COMMON/Packer set debug_ 0

UW/IP/Packer set SAddr_Bits 8
UW/IP/Packer set DAddr_Bits 8
UW/IP/Packer set debug_ 0
...
```

Create the proto stack

Remember to set the EvoLogics drivers as physical layer

```
...

set modem_ [new Module/UW/UwModem/EvoLogicsS2C]

# insert the module(s) into the node
$node_ addModule 8 $app_ 1 "UWA"
$node_ addModule 7 $transport_ 1 "UDP"
$node_ addModule 6 $routing_ 1 "IPR"
$node_ addModule 5 $ipif_ 1 "IPIF"
$node_ addModule 4 $mll_ 1 "ARP"
$node_ addModule 3 $mac_ 1 "ALOHA"
$node_ addModule 2 $uwal_ 1 "UWAL"
$node_ addModule 1 $modem_ 1 "S2C"
```

Link network addresses

Remember to link network entities in the routing tables and port destinations

```
$app_ set destAddr_ ...
$app_ set destPort_ ...


$routing_ addRoute <...> <...>
$mll_ addentry  <...> <...>
```

The devices

The EvoLogics S2C modems that are being used are emulated.

Each emulator provides 10 emulated devices, complete with all the features of real products.

To use them, it is necessary to connect to the EvoLogics VPN, where the modems are available through static addresses.

EvoLogics VPN

Requirement: OpenVPN and netcat

install with `apt install openvpn nc`

Connect to the vpn using the provided sign files:

```
openvpn --config #file.ovpn
```

Ping a modem to ensure connectivity (addr: 1-10):

```
ping 10.42.#NUM.1
```

Connect to a modem:

```
nc 10.42.#NUM.1 9200
```

Useful AT commands

+++ATC: change to *command mode*
required if the modem responds with smth as +++DROPCNT,…,…,…

AT?L: get current source level (=tx power)

AT!Ln: set source level (0-3, with 0 max.)

AT?AL: get local address

AT?AR: get remote address

ATZn: reset device (0-4, with 0 stored settings and restart)

TCL examples

From desert_samples/PHY/EvoLogics/:

- `test_uwevologics_application.tcl`:
  complete simulation for transmitting data through a socket in the APP layer
- `S2C_uwtdma.tcl`:
  simulation that sends random generated data (CBR in app layer) and uses TDMA in MAC layer

- To use the modem, be sure to be connected to the VPN and provide the correct address and port (10.42.#NUM.1 and 9200)