UNIVERSITÀ DI PISA

Distributed Systems and Middleware Technologies

JANET Home Simulator Installation Guide

Table of Contents

0) Prerequisites 1
1) Install Erlang/OTP 24.1 1
2) Install Rebar3 2
3) Configure the JANET Home Simulator 3
4) Start the JANET Home Simulator 4
5) Remote Nodes Hosts Configuration 5

0) Prerequisites

1. Linux operating system (*might* work on MacOs, not on Windows)
2. Availability of the *apt* package manager
3. Availability of *tar* or a similar archive manager

1) Install Erlang/OTP 24.1

Reference: Building and Installing Erlang/OTP

Since the precompiled Erlang/OTP package in the *apt* repository is typically outdated, it must be compiled and installed from its source code.

1. Ensure that the minimal dependencies required to compile and build Erlang are installed

```
yourUser@yourHost~$ sudo apt-get install make libncurses5-dev gcc perl sed openssl flex
```

2. Download the Erlang/OTP 24.1 source code archive from here.

```
yourUser@yourHost~$ wget https://erlang.org/download/otp_src_24.1.tar.gz
```

3. Unpack the archive and move to its unpacked directory

```
yourUser@yourHost~$ tar -zxf otp_src_24.1.tar.gz
yourUser@yourHost~$ cd otp_src_24.1
```

4. Set the $ERL_TOP environment variable to the current directory

```
yourUser@yourHost~/otp_src_24.1$ export ERL_TOP=`pwd`     # NOTE: ` not ‘
```

5. Configure the Erlang installation

```
yourUser@yourHost~/otp_src_24.1$ ./configure
```

Note that the command will report that some optional libraries (such as *fop*, *xmlint*, *wxWidgets*, *jinterface*, *odbc*), but they are not required for running the JANET Simulator.

6. Build the Erlang binaries

```
yourUser@yourHost~/otp_src_24.1$ make -j NUM_THREADS  # -j NUM_THREADS for parallel build
```

7. *(optional)* Validate the binaries by running a predefined test suite

```
yourUser@yourHost~/otp_src_24.1$ make release_tests
yourUser@yourHost~/otp_src_24.1$ cd release/tests/test_server
yourUser@yourHost~/otp_src_24.1/release/tests/test_server$ $ERL_TOP/bin/erl -s ts install
-s ts smoke_test batch -s init stop
```

Test results can be found in the “$ERL_TOP/release/tests/test_server/index.html” file (they should all be passed).

8. Install Erlang (might require root privileges depending on the default installation directory)

```
yourUser@yourHost~/otp_src_24.1$ [sudo] make install
```

9. Verify that Erlang was successfully installed and added to the $PATH via the *erl* command

```
yourUser@yourHost~$ erl
Erlang/OTP 24 [erts-12.1] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:1] [jit]

Eshell V12.1  (abort with ^G)
1> _
```

2) Install Rebar3

Reference: Rebar3 – Getting Started

Rebar3 is the de-facto standard project management tool for Erlang applications (akin to Maven for Java) which, by using a different project folder structure from the Erlang/OTP official one and by imposing a number of development constraints, often undocumented, represents a natural barrier for a more widespread adoption of the Erlang language.

1. Download the Rebar3 escript from here.

```
yourUser@yourHost~$ wget https://s3.amazonaws.com/rebar3/rebar3
```

2. Install Rebar3

```
yourUser@yourHost~$ ./rebar3 local install
===> Extracting rebar3 libs to ~/.cache/rebar3/lib...
===> Writing rebar3 run script ~/.cache/rebar3/bin/rebar3...
===> Add to $PATH for use: export PATH=$PATH:~/.cache/rebar3/bin
```

3. Permanently add the Rebar3 local installation folder ("~/.cache/rebar3/bin") to the $PATH

```
yourUser@yourHost~$ vi .bashrc
…
export PATH=$HOME/.cache/rebar3/bin:…:$PATH
```

4. Verify that Rebar3 has been correctly installed and added to the $PATH

```
yourUser@yourHost~$ rebar3
Rebar3 is a tool for working with Erlang projects.

  Set the environment variable DEBUG=1 for detailed output.

Usage: rebar3 [-h] [-v] [<task]
…
```

3) Configure the JANET Home Simulator

1. Install the JANET Home Simulator local configuration by executing the “install_config” bash script in its root folder

```
yourUser@yourHost~/janet_home_simulator$ ./install_config
JANET Home Simulator configuration successfully installed
```

   Note that this step is required to allow users to have their personal configurations without them being tracked and committed by git.

2. Configure the JANET Home Simulator by editing the “JANET Simulator Public Configuration Parameters” in the “config/sys.config" file, which are summarized here:

Parameter	Description	Allowed Values	Note
sim_rest_port	The OS port to be used by the JANET Simulator REST server	integer() > 0	-
remote_rest_server_addr	The address of the remote server accepting REST requests from JANET controller nodes	list() / string()	If deployed on the same host, the machine name (e.g. “yourHost”) must be used over “localhost”
remote_rest_server_port	The port of the remote server accepting REST requests from the JANET Controller nodes	integer() > 0	-
remote_rest_server_path	The remote REST server path where to send device state and connectivity updates	list() / string()	-
nodes_hosts	The list of hostnames JANET nodes can be deployed in	[list() / string()]	If the JANET Simulator host is to be included, use the full machine name (e.g. “yourHost”) over “localhost”

4) Start the JANET Home Simulator

The JANET Simulator Erlang Run-Time System (ERTS) can be started by executing the "start" bash script located in the "janet_home_simulator" folder, which will automatically fetch all application dependencies, compile them, and start the virtual machine and a set of application components (not including the JANET Simulator application itself).

```
yourUser@yourHost~/janet_home_simulator$ ./start
===> Verifying dependencies...
===> Analyzing applications...
===> Compiling ranch
===> Compiling cowlib
===> Compiling jsone
===> Compiling gun
===> Compiling cowboy
===> Analyzing applications...
===> Compiling janet_simulator
===> Compiling janet_controller
===> Compiling janet_device
Erlang/OTP 24 [erts-12.1] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:30] [jit]

Eshell V12.1  (abort with ^G)
(janet-simulator@yourHost)1> ===> Booted mnesia
_
```

The JANET Simulator application itself can be started with the "jsim:run()" command, which in its first execution will install the Mnesia database used by the application (which will initially be empty).

```
Erlang/OTP 24 [erts-12.1] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:30] [jit]

Eshell V12.1  (abort with ^G)
(janet-simulator@yourHost)1> ===> Booted mnesia

(janet-simulator@yourHost)1> jsim:run().
The Mnesia database is not installed, installing it now
JANET Simulator Mnesia database successfully installed
[locs_init]: <WARNING> No location is present in the database, no location tree will be
started
ok
_
```

The JANET Simulator application and ERTS can be stopped respectively via the "jsim:stop()" and "jsim:shutdown()" commands, with the complete set of supported commands being outlined in the JANET Home Simulator User Guide.

```
(janet-simulator@yourHost)2> jsim:stop().

JANET Simulator stopped
ok
(janet-simulator@yourHost)3> jsim:shutdown().
ok
(janet-simulator@yourHost)4> yourUser@yourHost~/janet_home_simulator$
```

5) Remote Nodes Hosts Configuration

The remote hosts JANET nodes can be deployed in (defined in the "nodes_hosts" configuration parameter) must have Erlang/OTP 24.1 installed (step 1) and require the following configuration steps:

1. Ensure the DNS reachability between the JANET Simulator and all remote hosts nodes.
   This can be obtained through the use of a DNS server or by manually mapping each host to its IP in the "/etc/hosts" files

```
yourUser@yourHost1~$ sudo vi /etc/hosts
192.168.0.1    yourHost1   # localhost
192.168.0.2    yourHost2
```

```
yourUser@yourHost2~$ sudo vi /etc/hosts
192.168.0.1    yourHost1
192.168.0.2    yourHost2   # localhost
```

2. Enable the JANET Simulator host to establish an *ssh* connection to each nodes host without being prompted for a password.
   This can be obtained via the following commands, with a more comprehensive guide being available here.

```
yourUser@JANETSimulatorHost~$ cd .ssh
yourUser@JANETSimulatorHost~/.ssh$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/yourUser/.ssh/id_rsa):        # Press Enter
Enter passphrase (empty for no passphrase):                              # Press Enter
Enter same passphrase again: [Pess enter key]                            # Press Enter
Your identification has been saved in /home/yourUser/.ssh/id_rsa.
Your public key has been saved in /home/yourUser/.ssh/id_rsa.pub.
The key fingerprint is:
33:b3:fe:af:95:95:18:11:31:d5:de:96:2f:f2:35:f9
yourUser@JANETSimulatorHost~/.ssh$ ssh-copy-id nodeHost1
yourUser@nodeHost1's password: ••••••   # Enter the "yourUser" password on "nodeHost1"
Number of key(s) added: 1

Now try logging into the machine, with : "ssh 'nodeHost1'"
And check to make sure that only the key(s) you wanted were added.
yourUser@JANETSimulatorHost~/.ssh$ ssh nodeHost1
yourUser@nodeHost1~$ _
```

3. Copy the folders containing the resource file and compiled bytecode of all the JANET Home Simulator application dependencies ("_build/default/lib") into the "lib" directory in the Erlang installation folder (default: "/usr/lib/erlang/lib") (might require root privileges).

```
yourUser@yourHost~/janet_home_simulator$ [sudo] cp -r _build/default/lib/cowlib
_build/default/lib/cowboy _build/default/lib/gun _build/default/lib/ranch
_build/default/lib/jsone _build/default/lib/janet_simulator
_build/default/lib/janet_controller _build/default/lib/janet_device /usr/lib/erlang/lib
yourUser@yourHost~/janet_home_simulator$
```