
black hat
ARSENAL
RPL Attacks Framework
Attacking RPL in WSNs
By Alexandre D'Hondt
#BHEU / @BLACK HAT EVENTS

Outline

- Introduction
- Background
- RPLA in a nutshell
- Managing simulations
- Building attacks
- Conclusion

Outline

- Introduction
  - Story
  - Scope
  - Objectives
- Background
- RPLA in a nutshell
- Managing simulations
- Building attacks
- Conclusion

Introduction > Story

- Catholic University of Louvain (BEL)
- LINGI2146 – *Mobile and Embedded Computing* [1]
- Started in March 2016
- Project team :

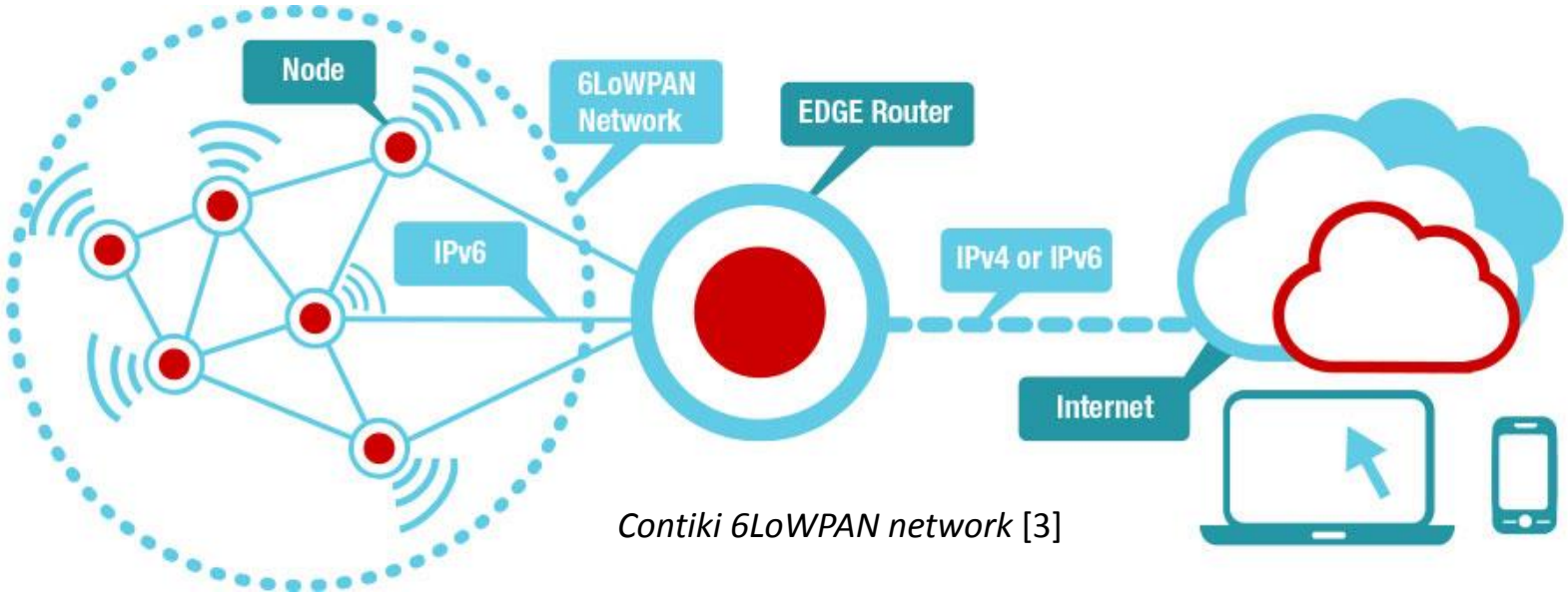
Prof. Ramin SADRE

Alex

Hussein

Introduction > Scope

- **W**ireless **S**ensor **N**etwork



Contiki 6LoWPAN network [3]

- RPL (RFC 6550) [2]

Application	CoAP
Transport	TCP/UDP
Network	6LoWPAN (IPv6/RPL)
Data Link	IEEE 802.15.4 MAC
Physical	IEEE 802.15.4 PHY

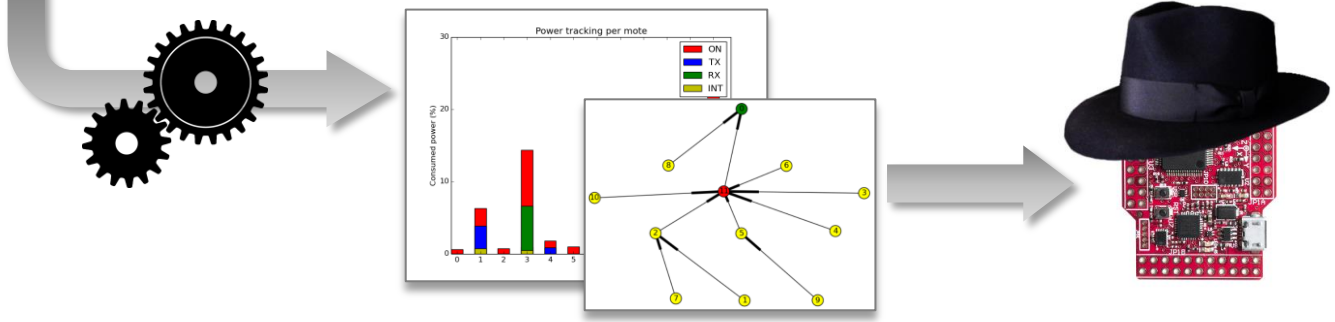
LLN protocol stack

- Contiki OS [4] + Cooja [5]

Introduction > Objectives

1. Tweak ContikiRPL
2. Build experiments
3. Automate simulations
4. Build a malicious sensor

```
{
  "sinkhole": {
    "simulation": {
      "title": "Sinkhole Attack",
      "number_motes": 10,
      "target": "z1",
      "duration": 3600,
      "debug": true
    },
    "malicious": {
      "type": "sensor",
      "constants": { ... },
      "external_library": "~/Projects/rpl-attacks/templates/rpl-modified"
    }
  },
  ...
}
```



Outline

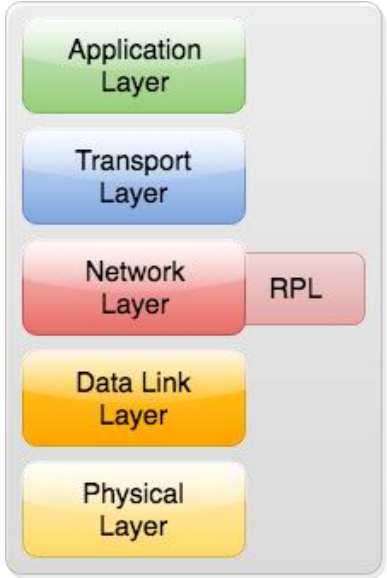
- Introduction
- Background
  - RPL
  - Contiki
  - Cooja
- RPLA in a nutshell
- Managing simulations
- Building attacks
- Conclusion

Background > RPL

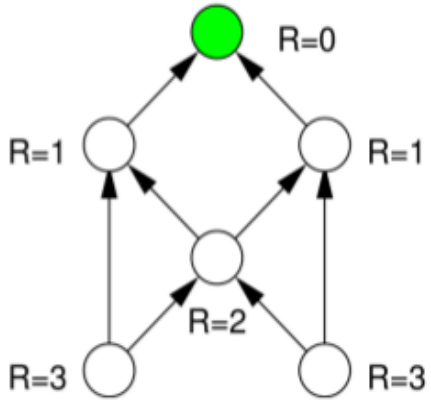
- IPv6 **R**outing **P**rotocol for **L**ow-Power and Lossy Networks [1]

  Characteristics :
  - Lossy, slow, unstable
  - Border router
  - Constrained (CPU, memory)
  - Large number of nodes

  Routing :
  - **D**estination **O**riented **D**irected **A**cyclic **G**raph
  - RPL instance = {disjoint DODAG's}
  - Node rank (0 = root)
  - DODAG version number
  - Control messages :
    - **D**AG **I**nformation **O**bject (periodically / on request)
    - **D**estination **A**dvertisement **O**bject (periodically)
    - **D**ODAG **I**nformation **S**olicitation (at join / rejoin / wakeup)



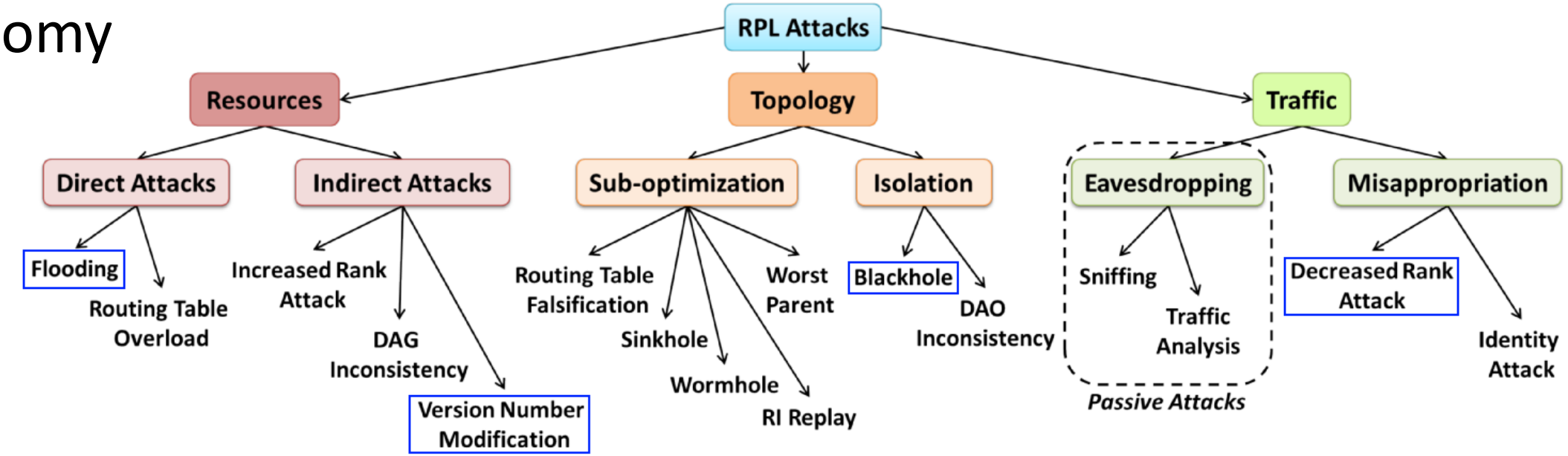
RPL in the protocol stack



DODAG example with ranks [1]

Background > RPL

- Taxonomy



Taxonomy of attacks against RPL networks [6]

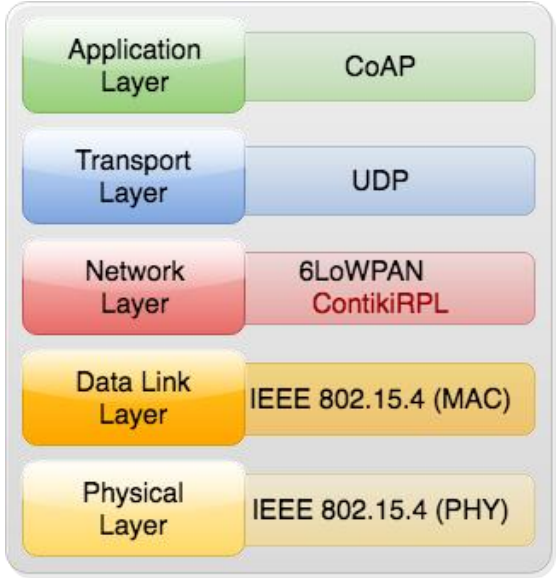
<u>Resources</u> : exhaustion of CPU, memory, energy
<u>Topology</u> : network disruption
<u>Traffic</u> : infiltration, information leakage

NB : Attacks framed in blue can be easilly implemented

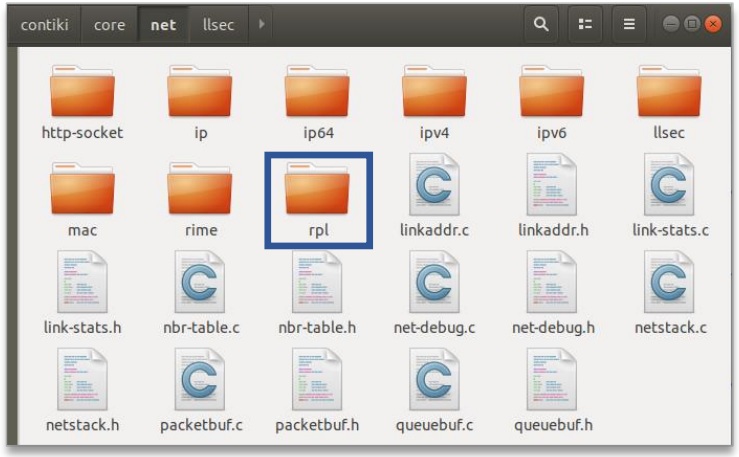
Background > Contiki

- Contiki OS [4]

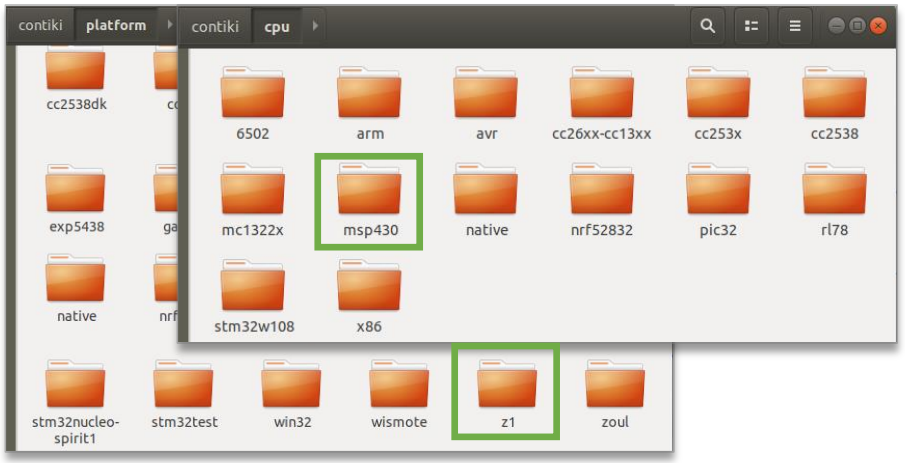
  Characteristics :
  - Open source
  - Memory-efficient
  - C programming
  - Tailored to small microcontroller Arch (e.g. MSP430)
  - Light network stack



Contiki's network protocol stack



Contiki's network library

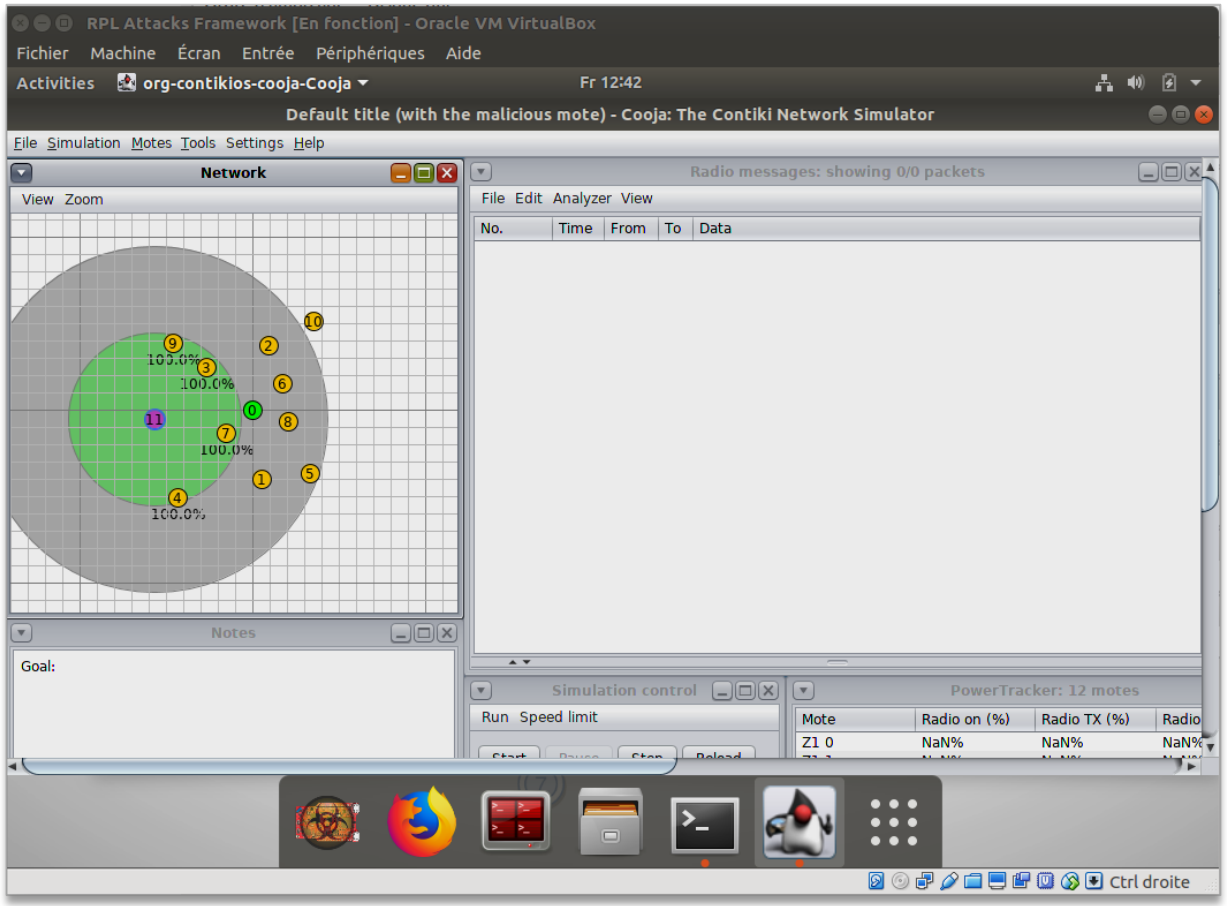


Contiki's implemented architectures

Background > Cooja

- Cooja [5]

  Characteristics :
  - Open source
  - Contiki's simulator
  - Java programming
  - Plugin architecture (apps)
  - User-friendly GUI
  - Data collection



black hat ARSENAL

Outline

- Introduction
- Background
- RPLAF in a nutshell
  - Basics
  - Demo 1 : Getting started
  - Demo 2 : Getting help
  - Demo 3 : Start the built-in demo
- Managing simulations
- Building attacks
- Conclusion

RPLAF in a nutshell > Basics

- Setup
  - Vagrant box
  - Manual installation

  https://rpl-attacks.readthedocs.io/en/latest/install/

- Usage

  https://rpl-attacks.readthedocs.io/en/latest/usage/

- Maintenance
  - Update of Contiki + RPL Attacks Framework
  - Crash report generation in experiment folder

  https://rpl-attacks.readthedocs.io/en/latest/commands/

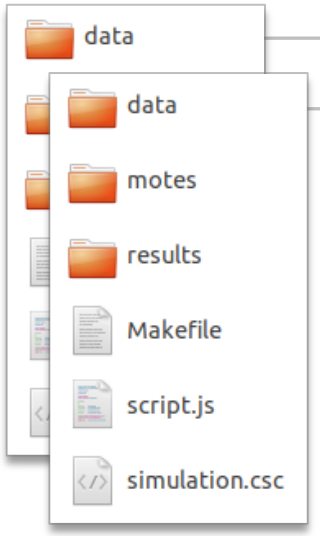
RPLAF in a nutshell > Basics

- **Definitions**
  - Simulation
  - Experiment
  - Campaign
  - Building block (BB)
  - Attack ({BB})

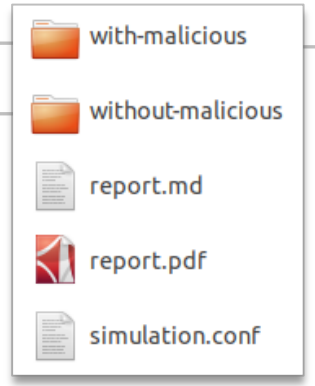
- **Items**

`simulation.csc`	Cooja simulation config file
`script.js`	Cooja simulation runtime script
`motes`	motes compiled by the framework
`data`	data collected by Cooja
`results`	data processed by the framework
`report.md`	experiment report in Markdown

Simulation



Experiment



Campaign

```
45   },
46   "version-number-attack": {
47     "simulation": {
48       "title": "Version Number Modification Attack",
49       "goal": "Demonstrate that this attack can ...",
50       "notes": "With its modified RPL file, the ...",
51       "comment-1": "As it can be seen, the DODAG...",
52       "comment-2": "By looking at the power cons..."
53     },
54     "malicious": {
55       "building-blocks": ["increased-version"]
56     }
57   },
58   "blackhole-attack": {
59     "simulation": {
```

Building blocks

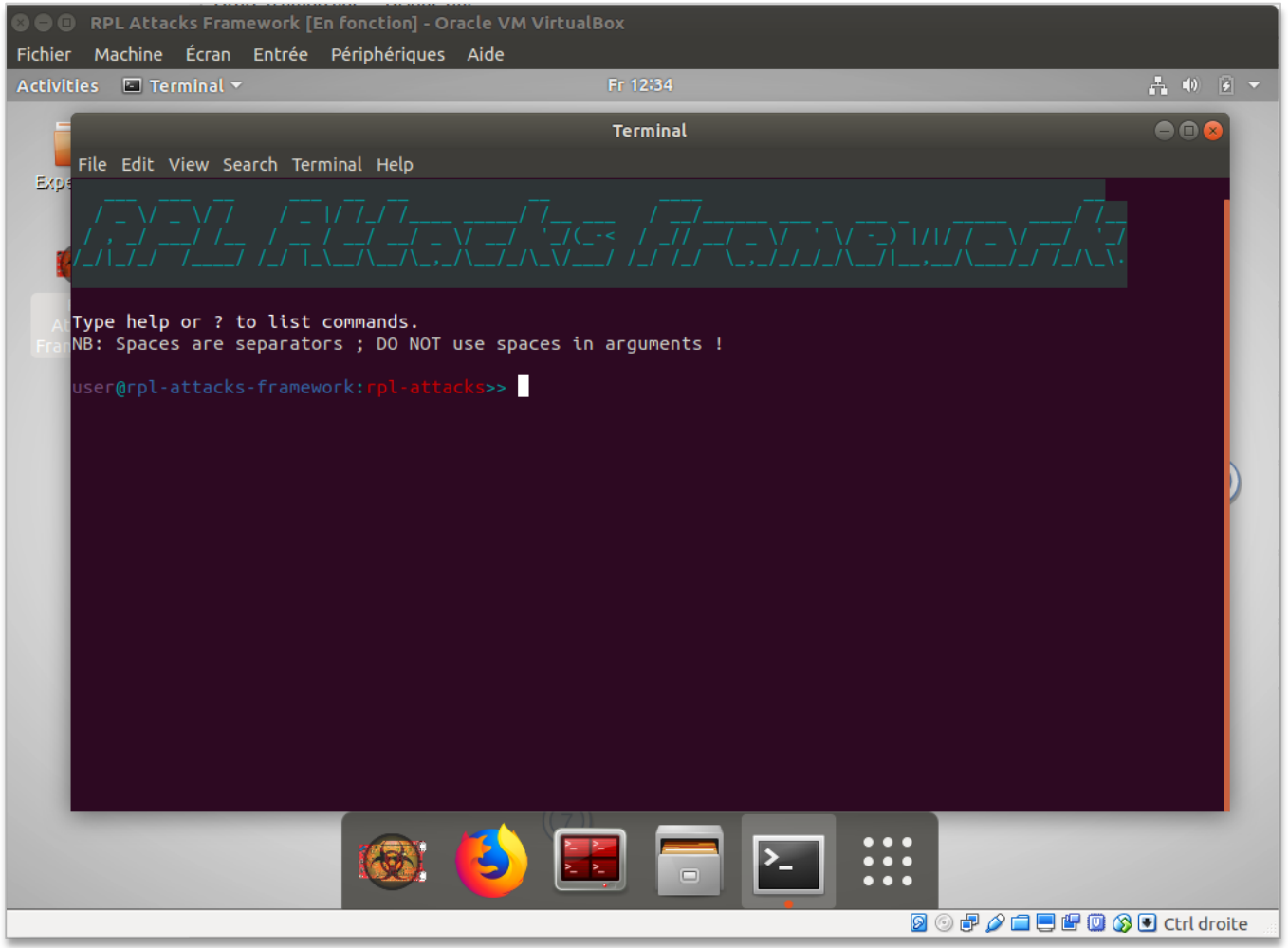
```
22 {
23   "hello-flood": {
24     "RPL_CONF_DIS_INTERVAL": 0,
25     "RPL_CONF_DIS_START_DELAY": 0,
26     "rpl-timers.c": ["next_dis++;", "next_dis++; int i=0; while (i<20) {i++; dis_ou
27   },
28   "increased-version": {
29     "rpl-icmp6.c": ["dag->version;", "dag->version++;"]
30   },
31   "decreased-rank": {
32     "RPL_CONF_MIN_HOPRANKINC": 0,
33     "rpl-private.h": [
34       ["#define RPL_MAX_RANKINC            (7 * RPL_MIN_HOPRANKINC)", "#define RPL
35       ["#define INFINITE_RANK                0xffff", "#define INFINITE_RANK 256
36     ],
37     "rpl-timers.c": ["rpl_recalculate_ranks();", null]
38   }
39 }
```

RPLAF in a nutshell > Demo 1 : Getting started

- Start : Vagrant box

- Actions :
  1) **Start the framework**
  2) Type "`update`"

- End State :
  ✓ Up-to-date RPLAF



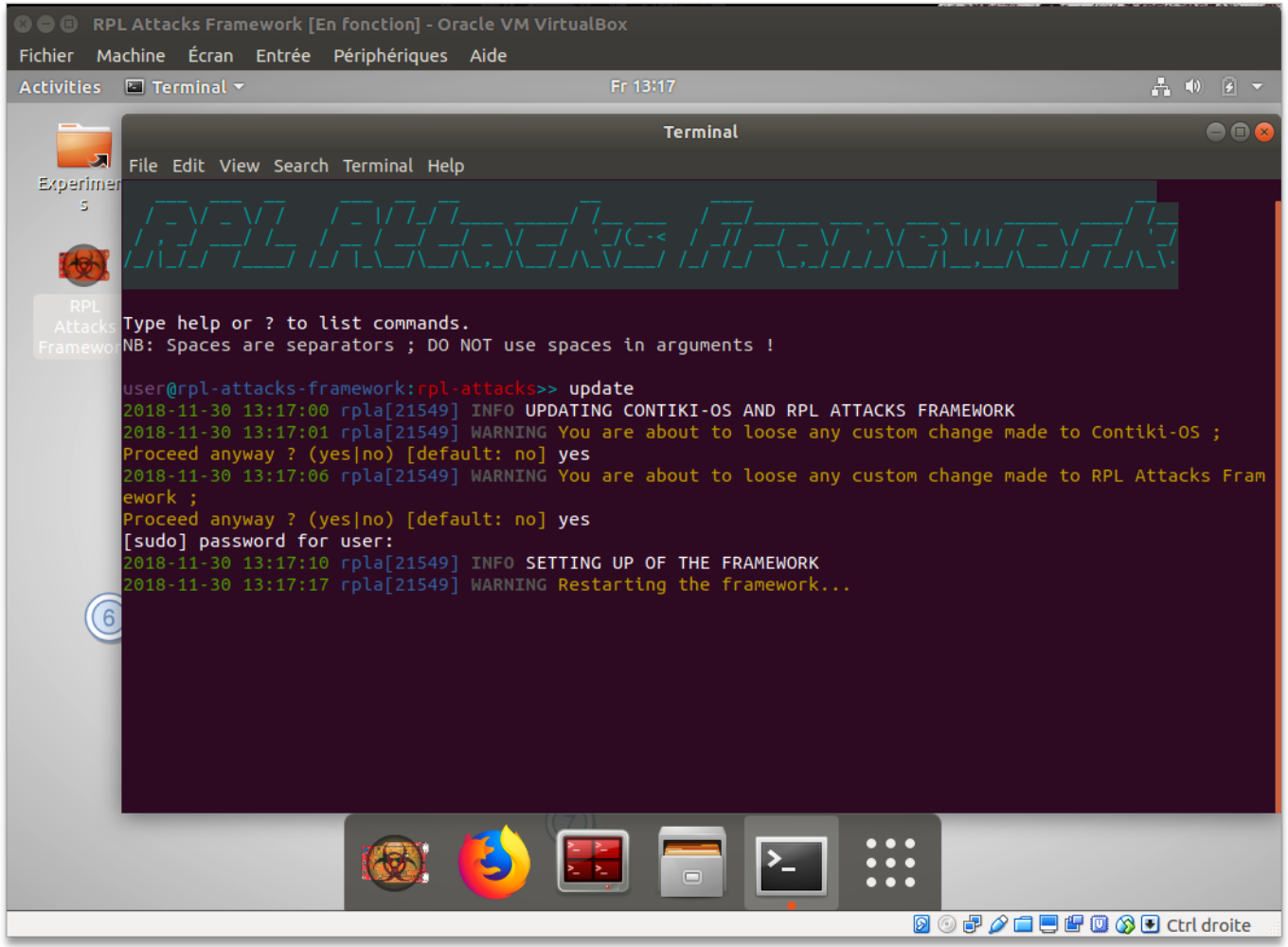
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

RPLAF in a nutshell > Demo 1 : Getting started

- Start : Vagrant box

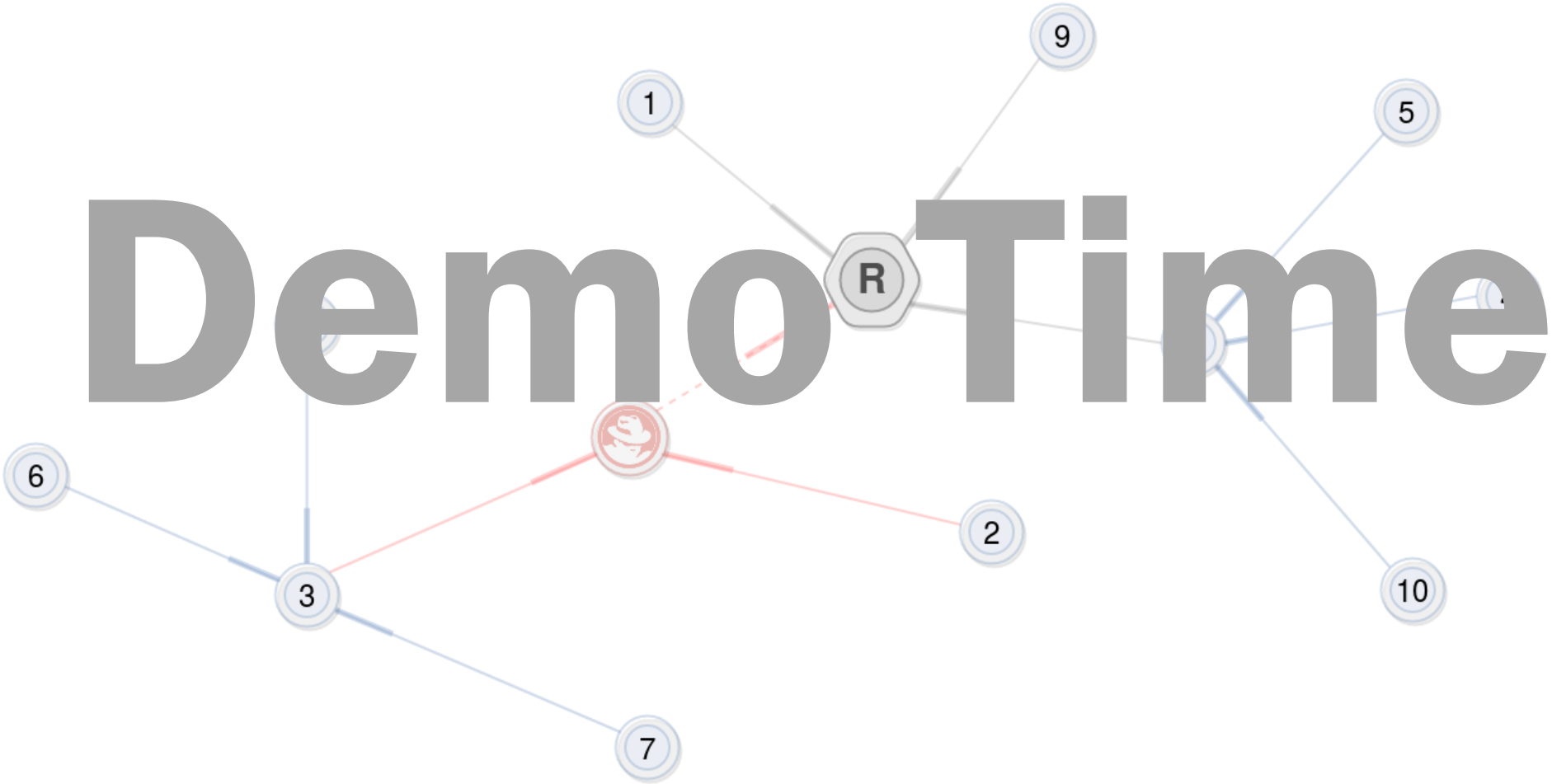
- Actions :
  1) Start the framework
  2) **Type "`update`"**

- End State :
  ✓ Up-to-date RPLAF



Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

RPLAF in a nutshell > Demo 1 : Getting started



Demo Time

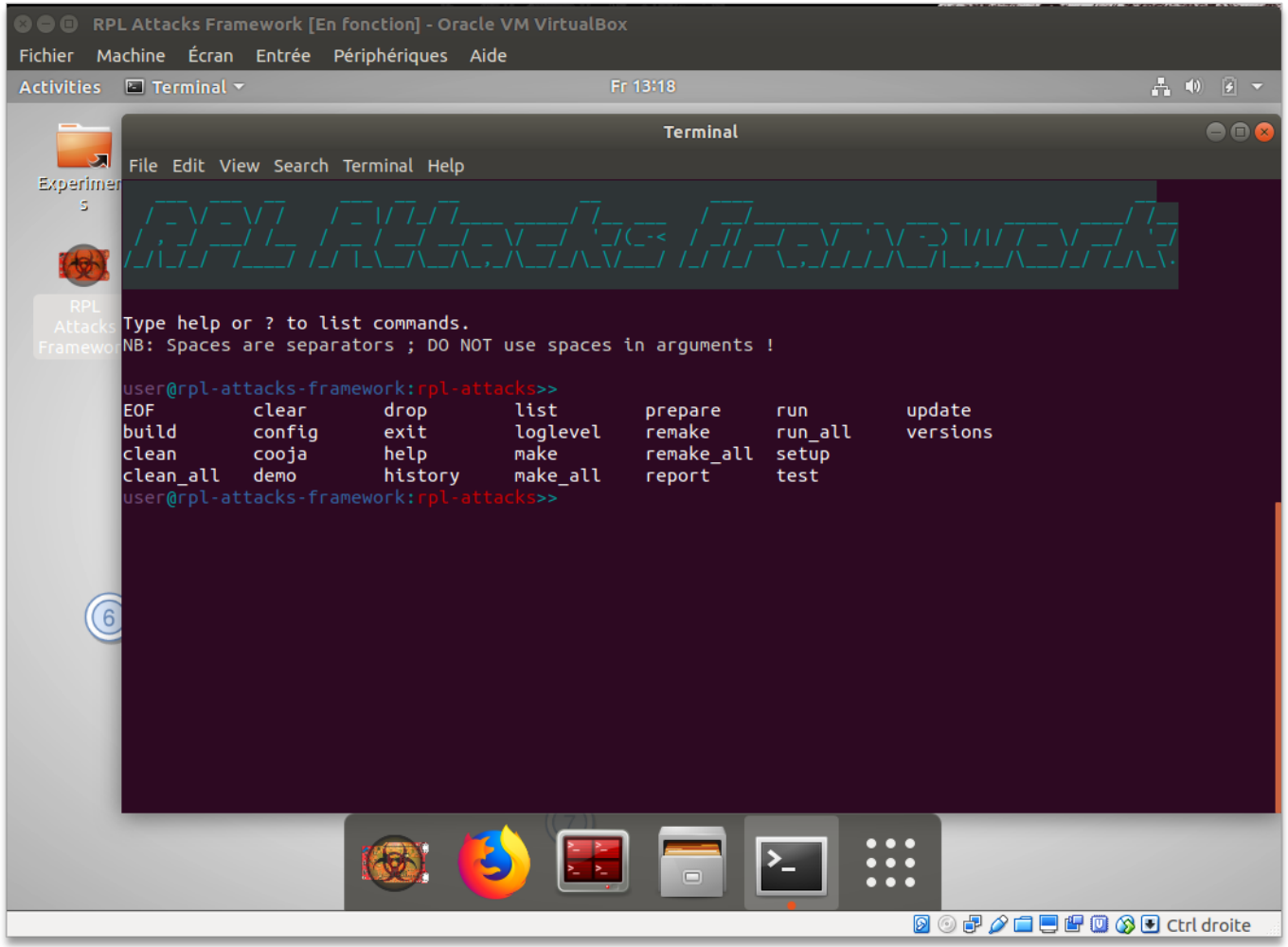
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

RPLAF in a nutshell > Demo 2 : Getting help

- Start : Vagrant box

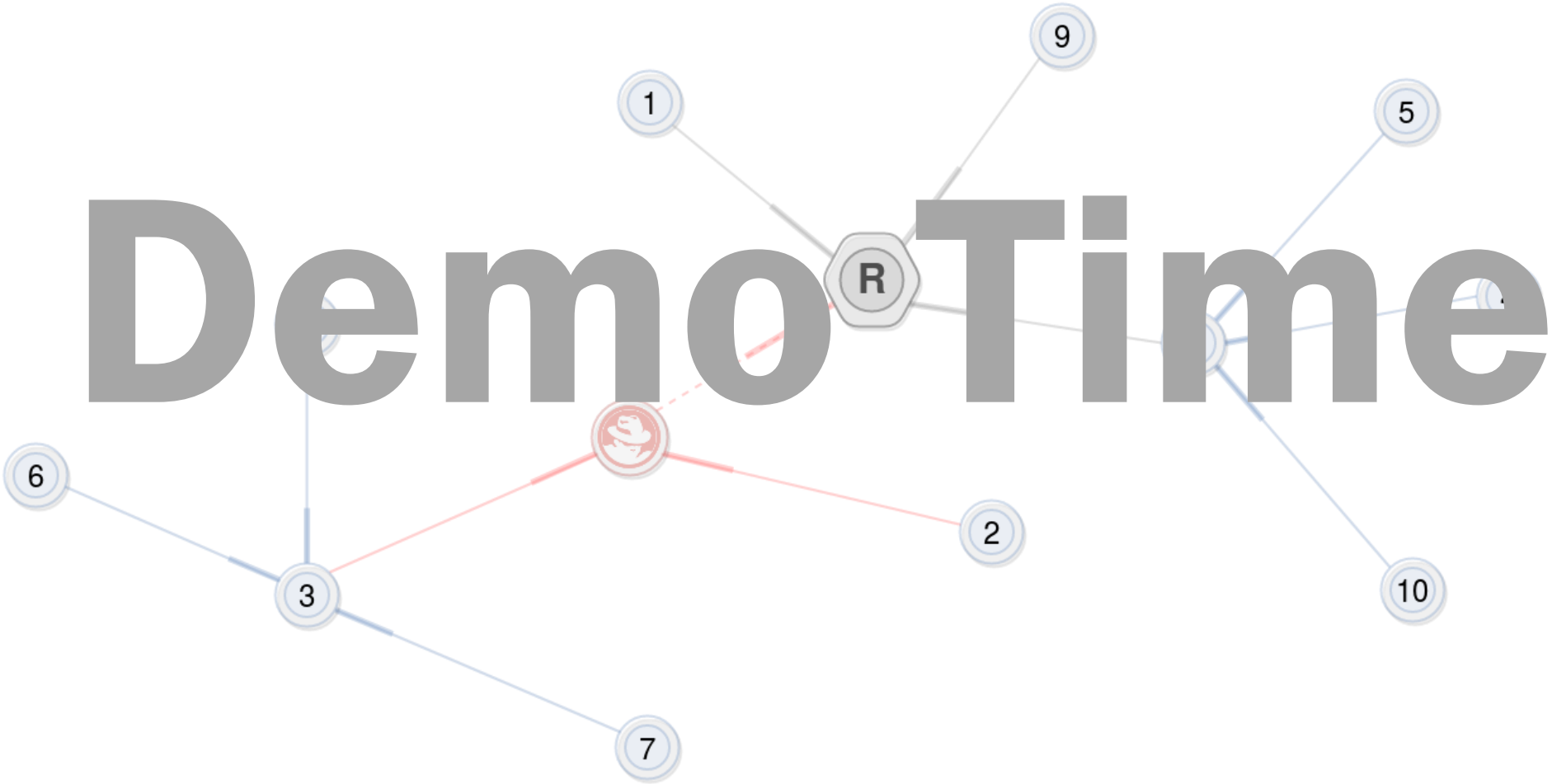
- Actions :
  1) **Type <TAB><TAB>**
  2) Type "`help`"
  3) Type "`help prepare`"

- End State :
  ✓ No change



Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

RPLAF in a nutshell > Demo 2 : Getting help



Demo Time

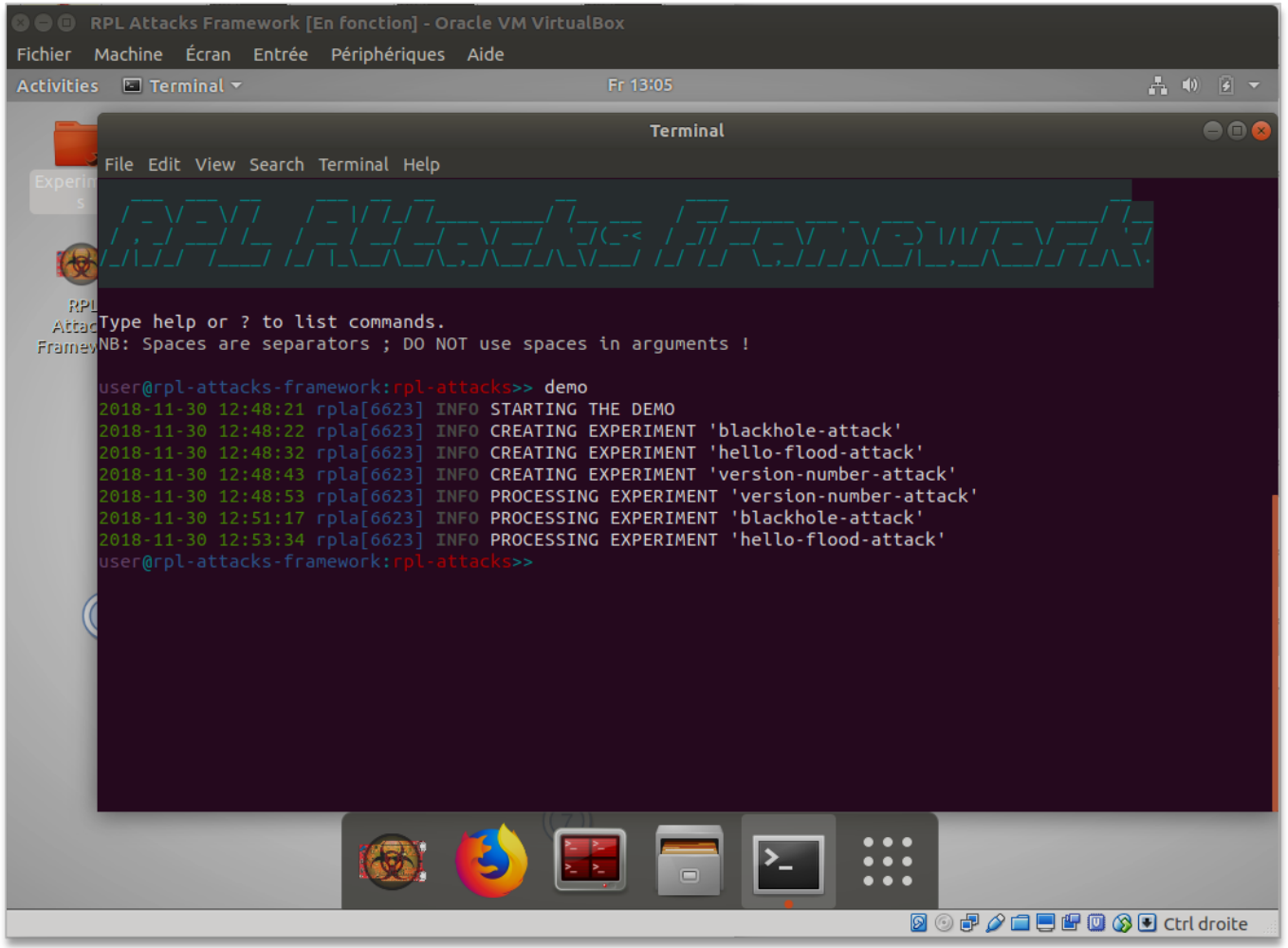
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

RPLAF in a nutshell > Demo 3 : Start the built-in demo

- Start : Framework started

- Actions :
  1) **Type "demo"**

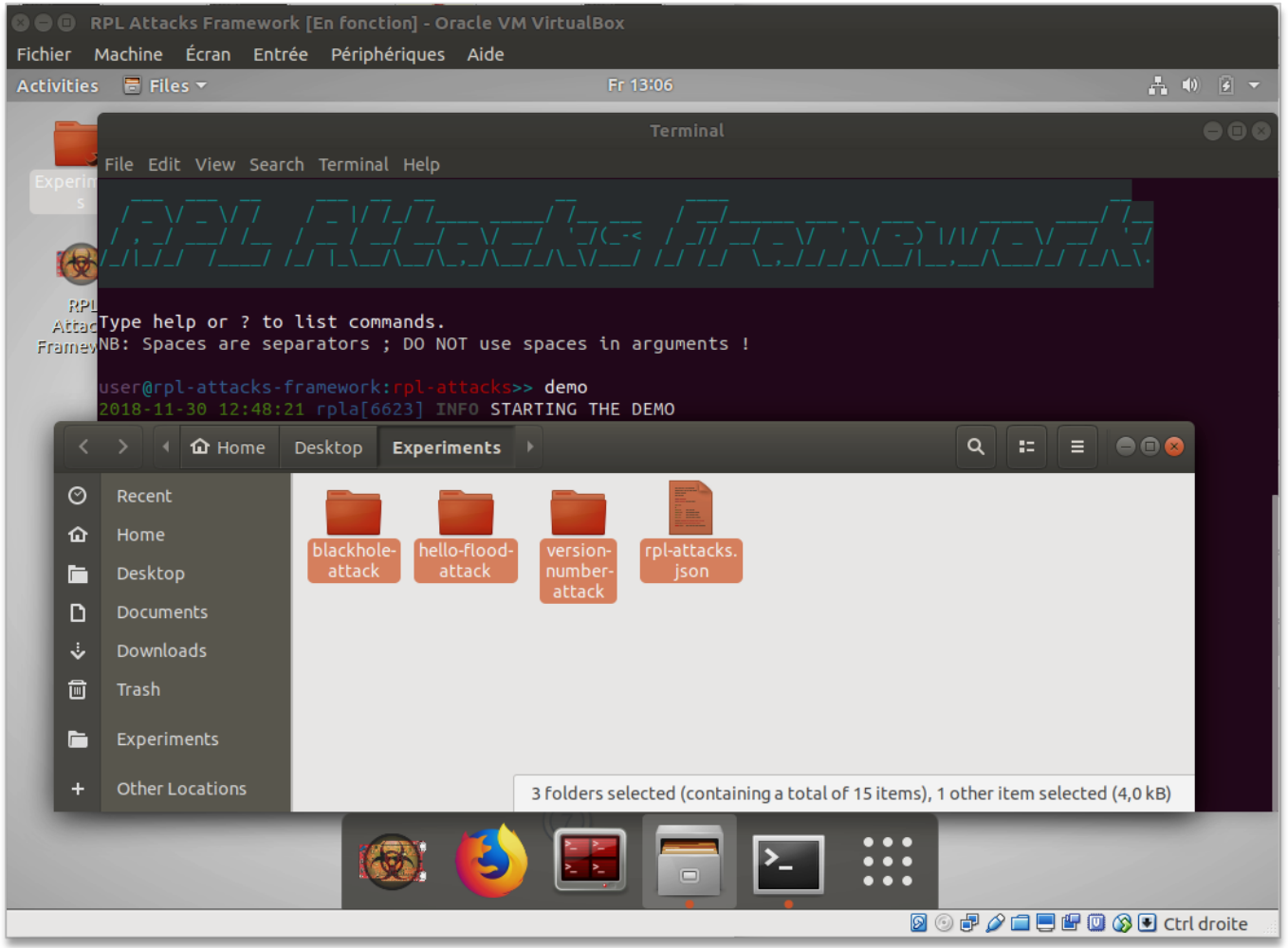
- End State :
  - ✓ Demo campaign created in experiments folder
  - ✓ Demo campaign run, reports ready



Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

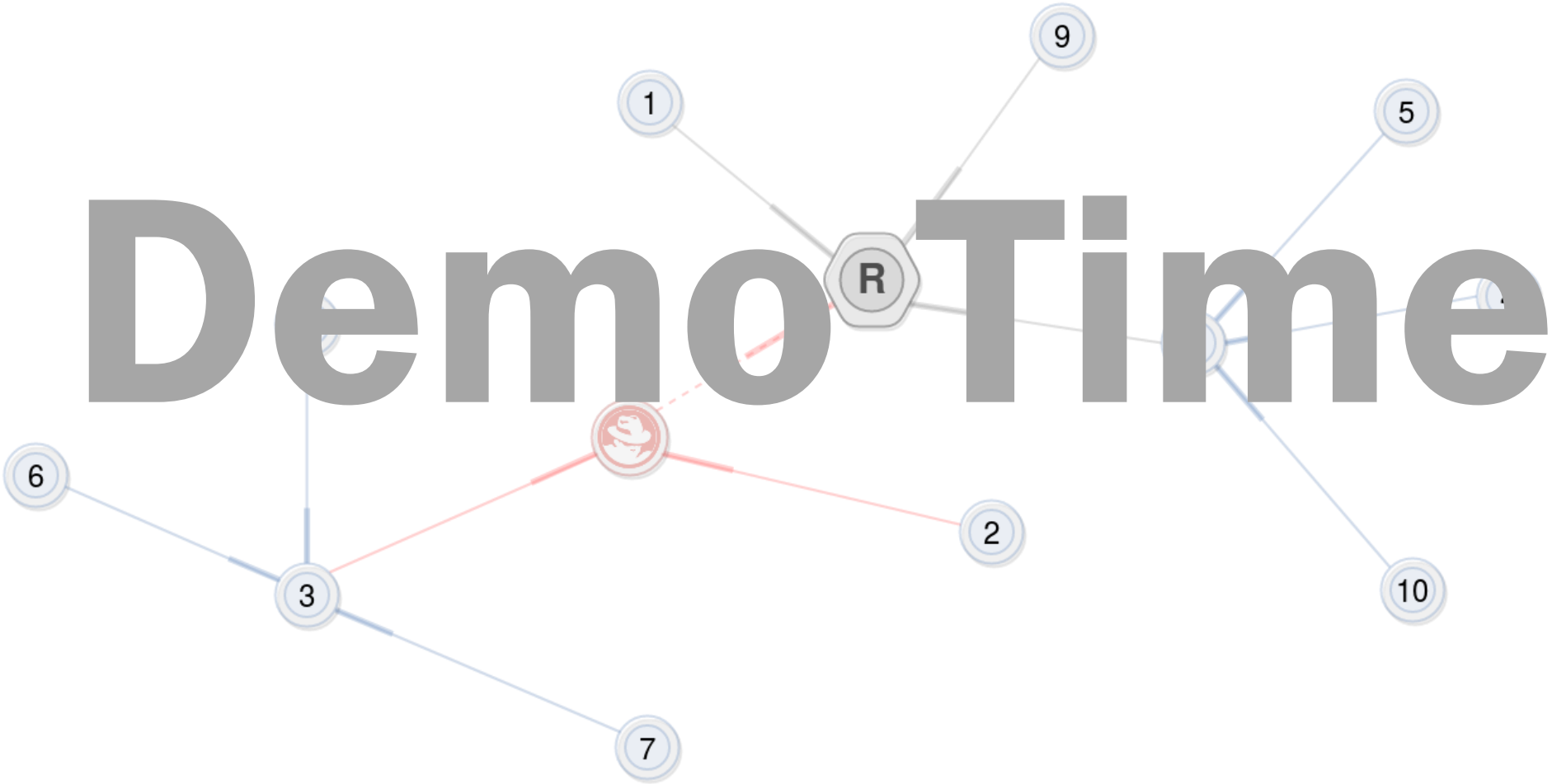
RPLAF in a nutshell > Demo 3 : Start the built-in demo

- Start : Framework started
- Actions :
  1) Type "demo"
- End State :
  - ✓ **Demo campaign created in experiments folder**
  - ✓ Demo campaign run, reports ready



Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

RPLAF in a nutshell > Demo 3 : Start the built-in demo



Demo Time

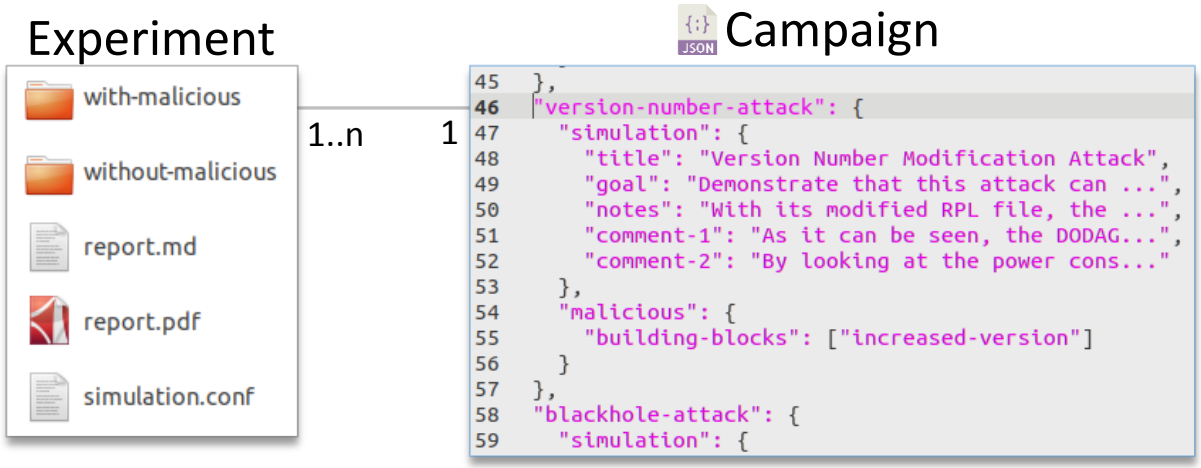
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Outline

- Introduction
- Background
- RPLA in a nutshell
- Managing simulations
  - Basics
  - Demo 4 : Make an experiment
  - Demo 5 : Make a campaign
  - Demo 6 : Tune a report
- Building attacks
- Conclusion

Managing simulations > Basics

- Entities



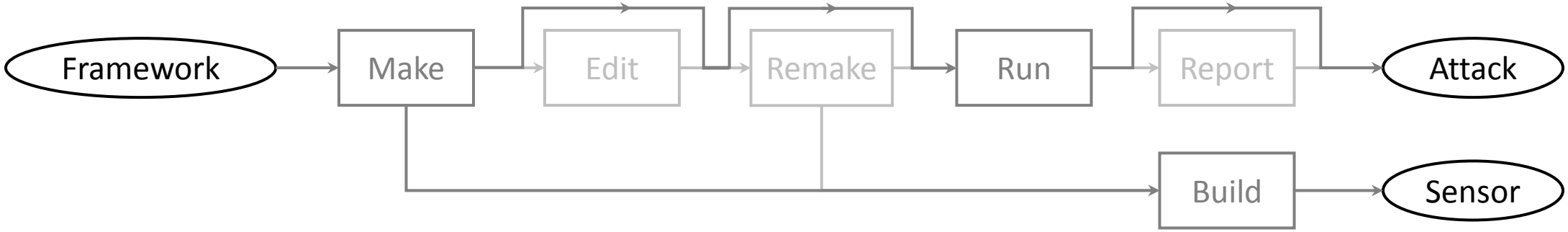
- Commands

	`prepare`	Make a new JSON campaign file
`make`	`make_all`	Make a new experiment / all experiments from a JSON campaign file
`cooja`		Edit the experiment's simulation with a malicious mote in Cooja
`remake`	`remake_all`	Remake the malicious mote for an experiment / all experiments from a campaign
`run`	`run_all`	Run an experiment / all experiments from a campaign
`report`		Generate the report of an experiment
`clean`	`clean_all`	Remove an experiment / all experiments from a campaign
`build`		Build the real sensor with the malicious mote of an experiment

Related documentation : https://rpl-attacks.readthedocs.io/en/latest/commands/

Managing simulations > Basics

- Process



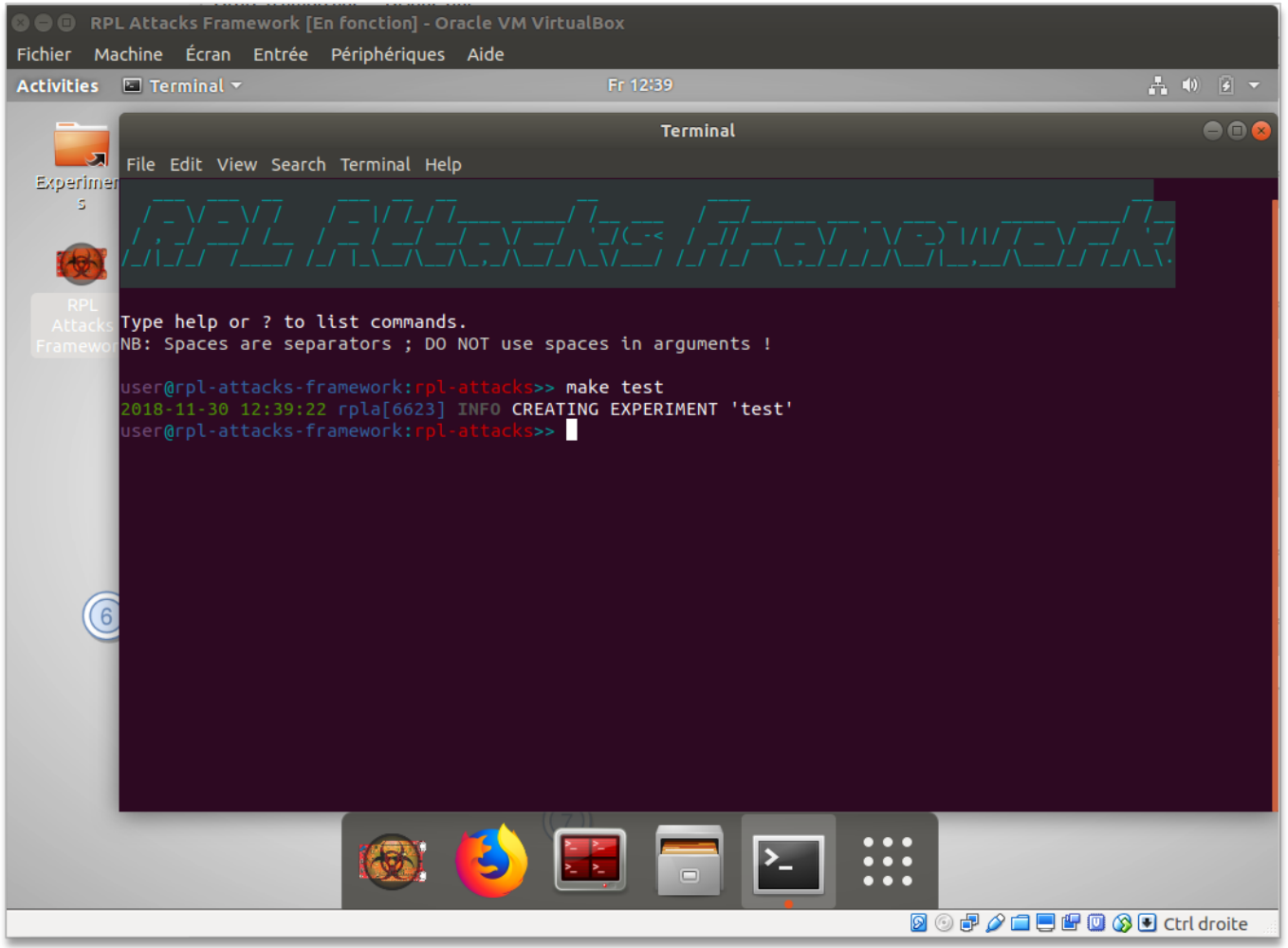
- Demonstrations
  4) Make + Edit + Run (single experiment)
  5) Make + Edit + Run (campaign)
  6) Report (single experiment)

Managing simulations > Demo 4 : Make an experiment

- Start : Framework started

- Actions :
  1) **Type "`make test`"**
  2) Type "`cooja test`"
  3) Type "`run test`"

- Output :
  - ✓ New `test` experiment created in experiments folder
  - ✓ New `test` experiment tuned
  - ✓ New `test` experiment run, report ready



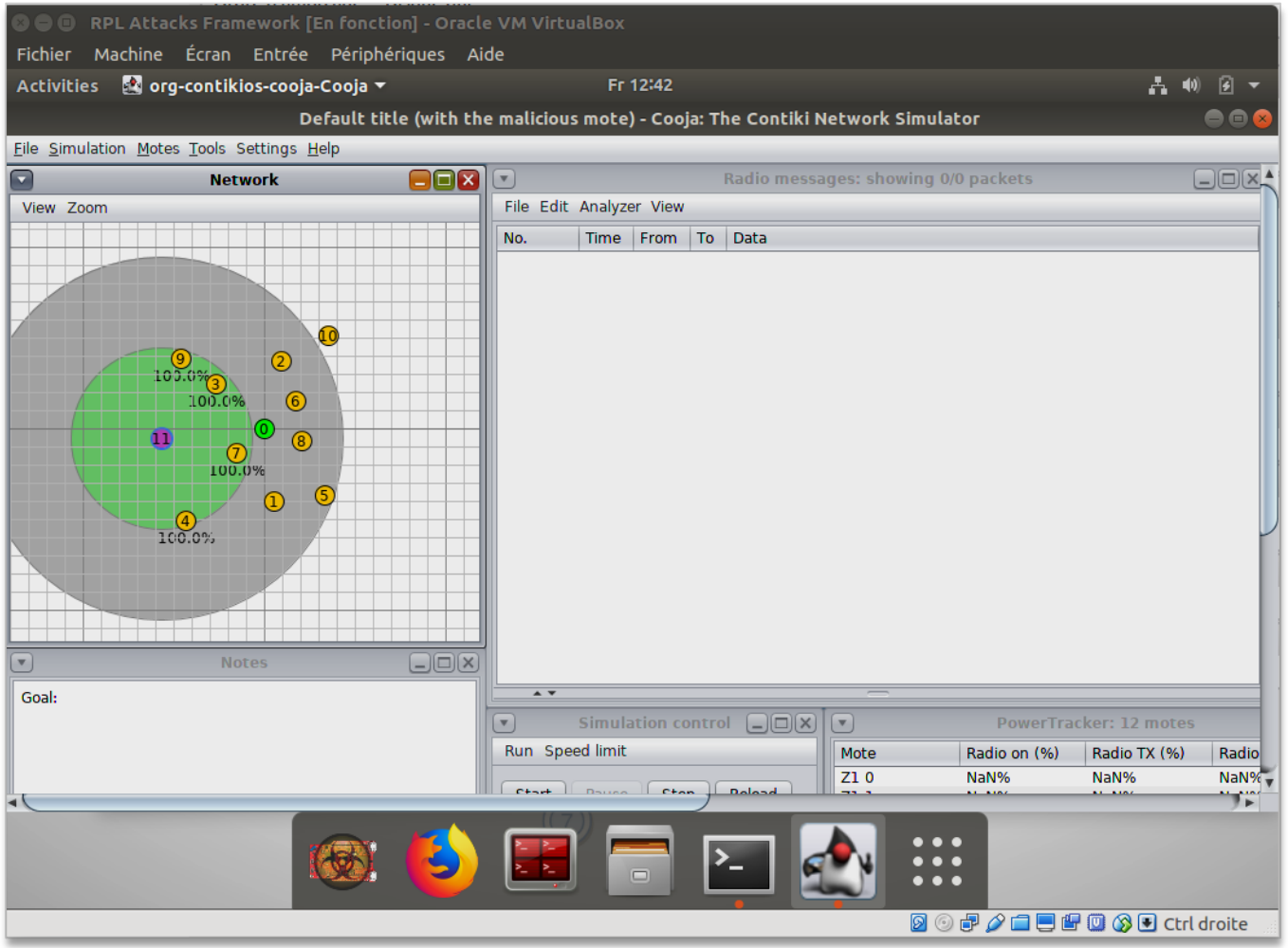
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Managing simulations > Demo 4 : Make an experiment

- Start : Framework started

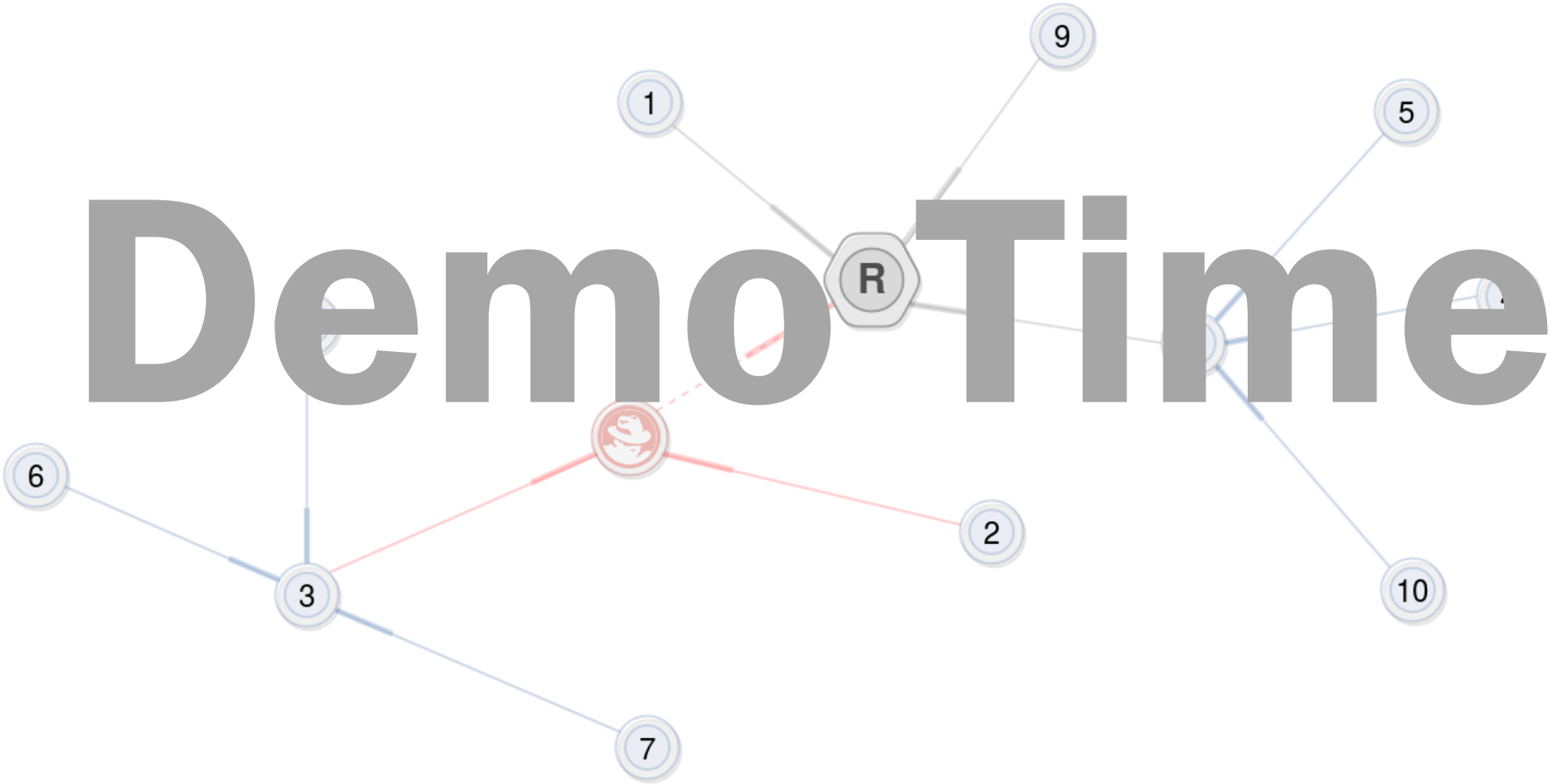
- Actions :
  1) Type "`make test`"
  2) **Type "`cooja test`"**
  3) Type "`run test`"

- Output :
  - ✓ New `test` experiment created in experiments folder
  - ✓ New `test` experiment tuned
  - ✓ New `test` experiment run, report ready



Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Managing simulations > Demo 4 : Make an experiment



Demo Time

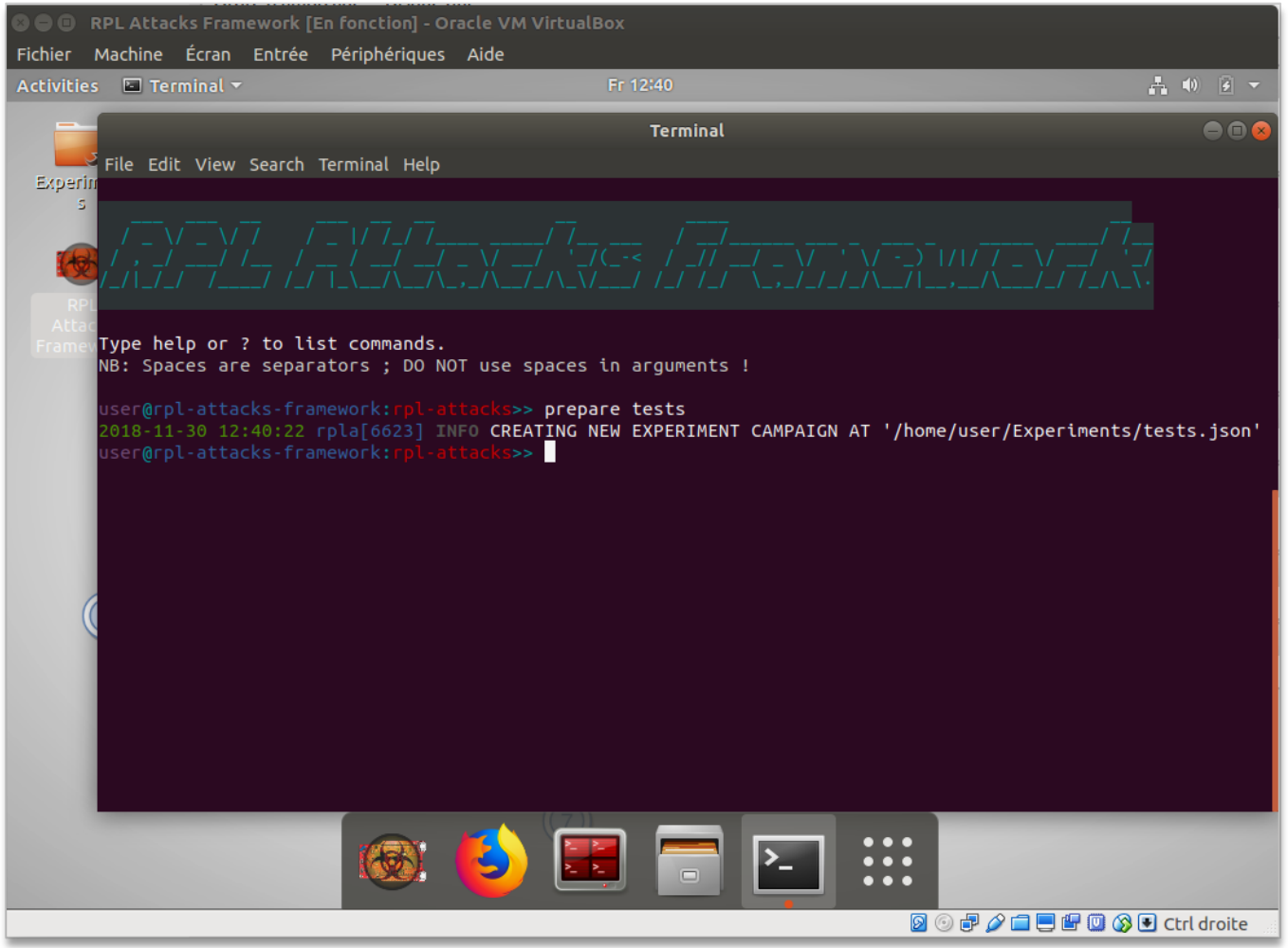
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Managing simulations > Demo 5 : Make a campaign

- Start : Framework started

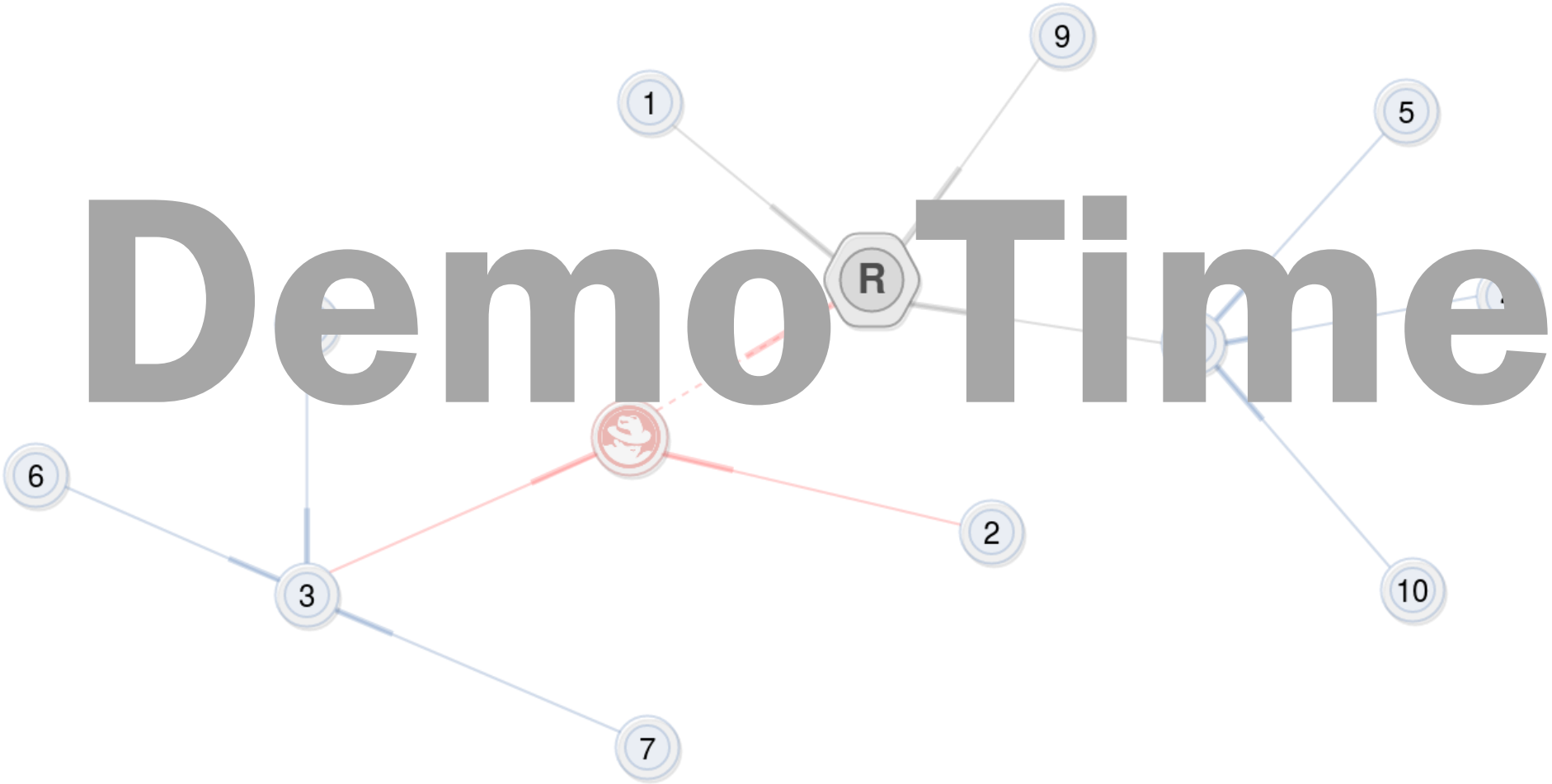
- Actions :
  1) **Type "`prepare tests`"**
  2) Edit `tests.json` (exp. folder)
  3) Type "`make_all tests`"
  4) Type "`run_all tests`"

- Output :
  - ✓ `tests` campaign created (exp. folder)
  - ✓ `tests` child exp. created (exp. folder)
  - ✓ `tests` child exp. run, reports ready



Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Managing simulations > Demo 5 : Make a campaign



Demo Time

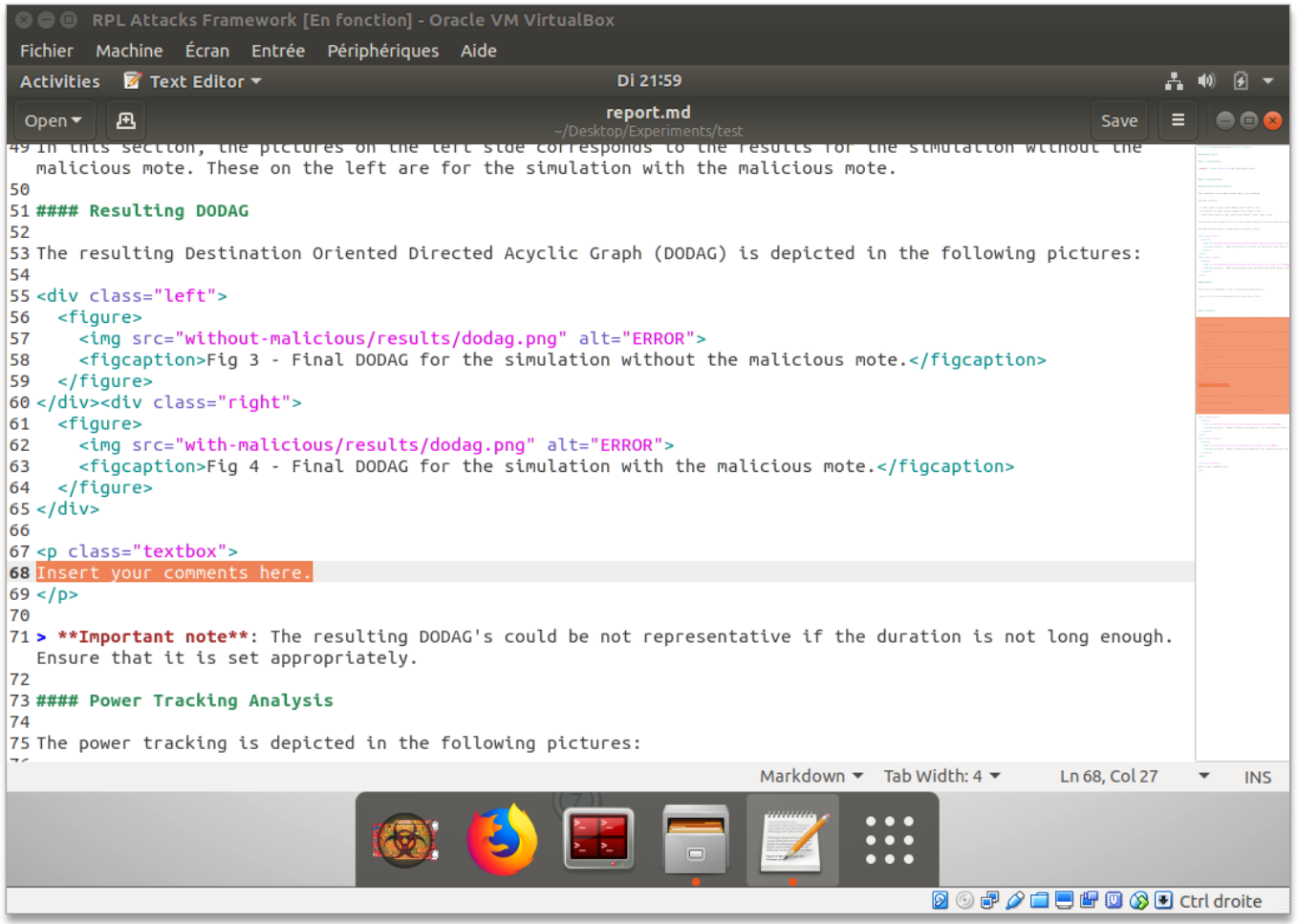
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Managing simulations > Demo 6 : Tune a report

- Start : `test` experiment run

- Actions :
  1) **Edit `report.md` (exp. folder)**
  2) Type "`report test`"

- Output :
  ✓ Report of `test` experiment tuned



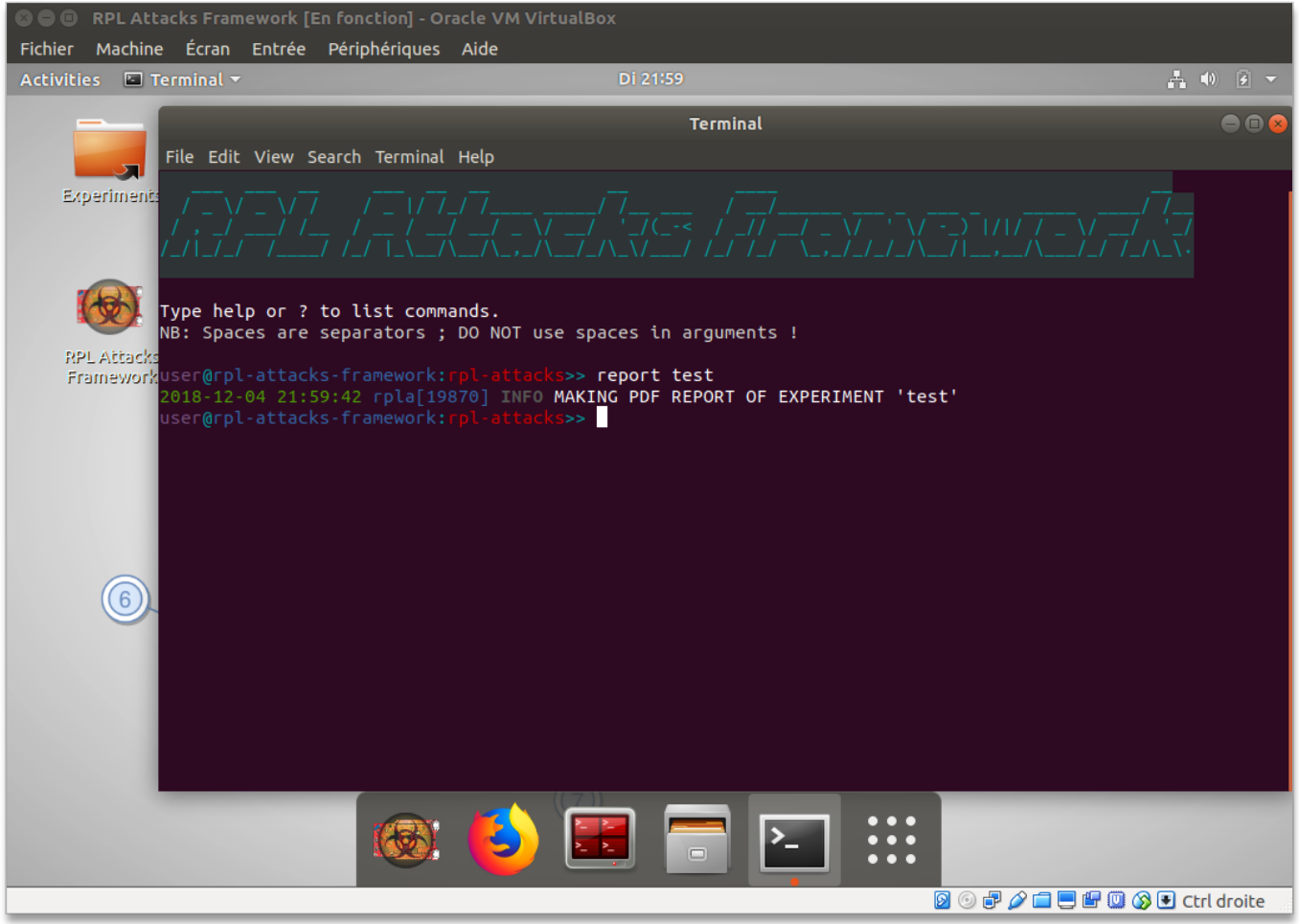
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Managing simulations > Demo 6 : Tune a report

- Start : `test` experiment run

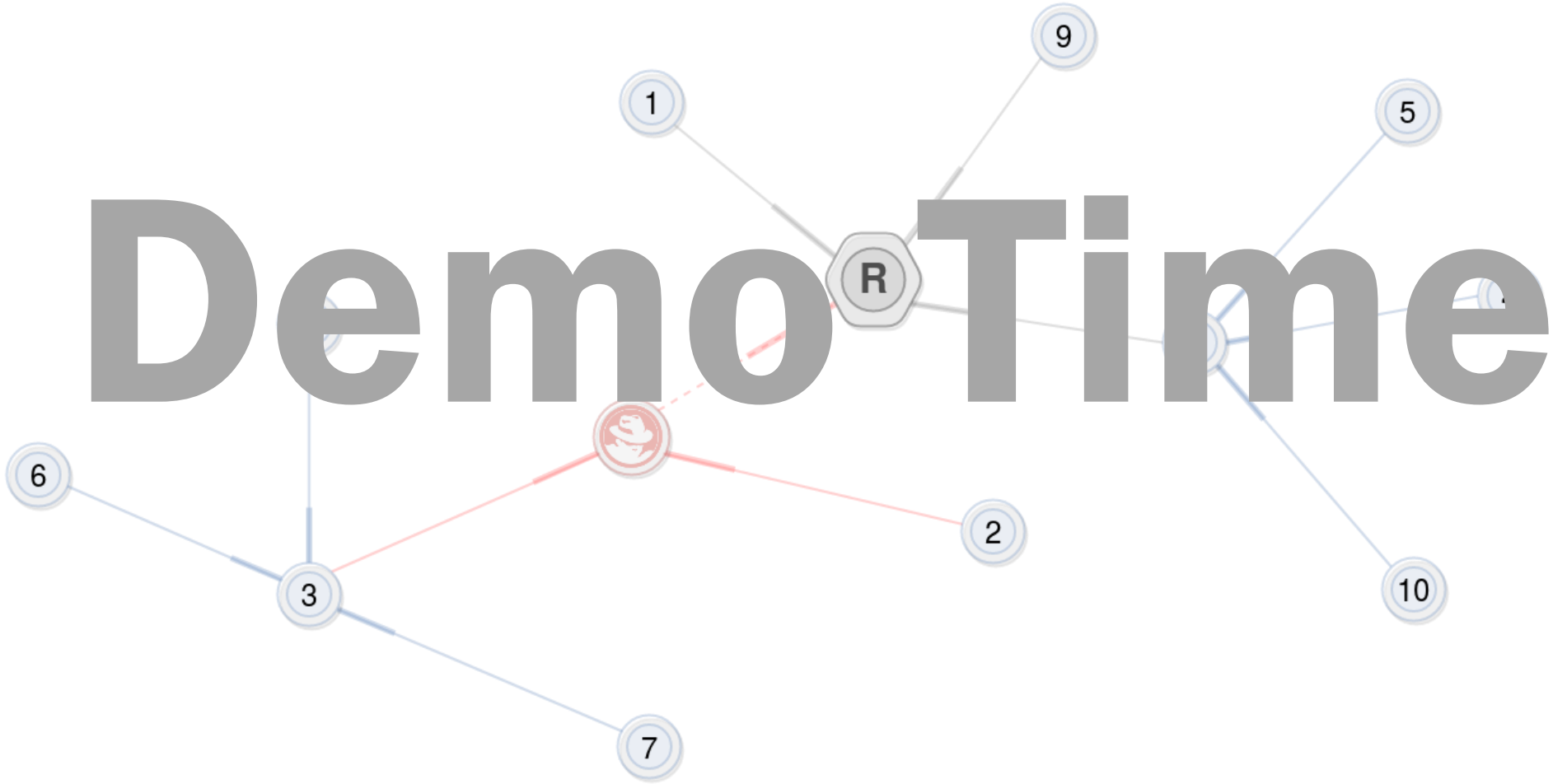
- Actions :
  1) Edit `report.md` (exp. folder)
  2) **Type "`report test`"**

- Output :
  ✓ Report of `test` experiment tuned



Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Managing simulations > Demo 6 : Tune a report



Demo Time

Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

black hat ARSENAL

Outline

- Introduction
- Background
- RPLA in a nutshell
- Managing simulations
- Building attacks
  - Basics
  - Demo 7 : Implement *Flooding*
  - Demo 8 : Implement *Version Number Modification*
  - Demo 9 : Implement *Decreased Rank*
- Conclusion

Building attacks > Basics

- Entities

- RPL alterations
  1. Modify constant
  2. Replace line(s)
  3. Use a modified RPL library

Building blocks

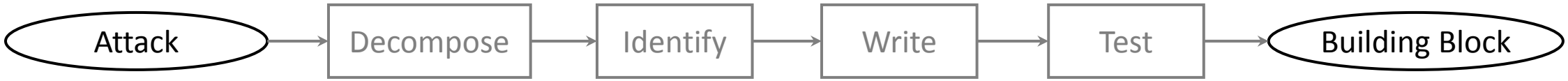
```
22 {
23   "hello-flood": {
24     "RPL_CONF_DIS_INTERVAL": 0,
25     "RPL_CONF_DIS_START_DELAY": 0,
26     "rpl-timers.c": ["next_dis++;", "next_dis++; int i=0; while (i<20) {i++; dis_ou
27   },
28   "increased-version": {
29     "rpl-icmp6.c": ["dag->version;", "dag->version++;"]
30   },
31   "decreased-rank": {
32     "RPL_CONF_MIN_HOPRANKINC": 0,
33     "rpl-private.h": [
34       ["#define RPL_MAX_RANKINC             (7 * RPL_MIN_HOPRANKINC)", "#define RPL
35       ["#define INFINITE_RANK                  0xffff", "#define INFINITE_RANK 256
36     ],
37     "rpl-timers.c": ["rpl_recalculate_ranks();", null]
38   }
39 }
```

Campaign

```
45   },
46   "version-number-attack": {
47     "simulation": {
48       "title": "Version Number Modification Attack",
49       "goal": "Demonstrate that this attack can ...",
50       "notes": "With its modified RPL file, the ...",
51       "comment-1": "As it can be seen, the DODAG...",
52       "comment-2": "By looking at the power cons..."
53     },
54     "malicious": {
55       "building-blocks": ["increased-version"]
56     }
57   },
58   "blackhole-attack": {
59     "simulation": {
```

Building attacks > Basics

- Process

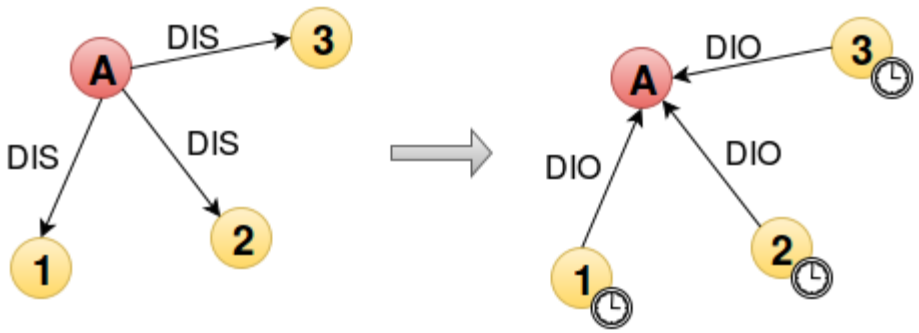


- Demonstrations
  7) All steps for *Flooding* attack
  8) All steps for *Version Number Modification* attack
  9) All steps for *Decreased Rank* attack

Related documentation : https://rpl-attacks.readthedocs.io/en/latest/building-blocks/

Building attacks > Demo 7 : Implement *Flooding*

- Attack :
  - DIS generation
  - Causes sensors to send DIO + reset trickle timer



- Decompose :
  - Only 1 BB required : HELLO flood

- Identify :
  - Depends on RPL constants `RPL_CONF_DIS_INTERVAL` and `RPL_CONF_DIS_START_DELAY`
  - `rpl-timers.c` handles DIS sending

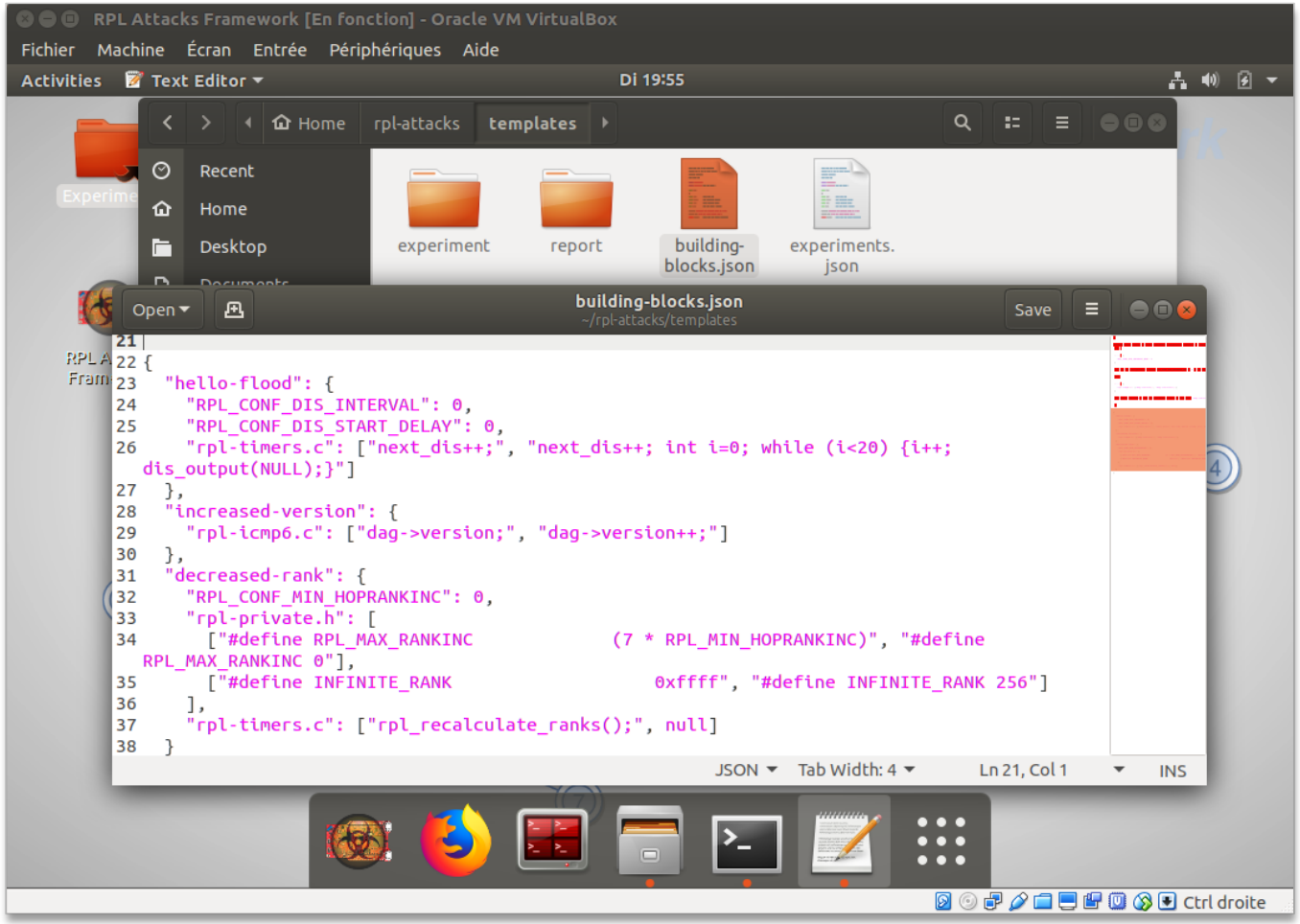
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 7 : Implement *Flooding*

- Start : Vagrant box

- Actions :
  1) **Edit `building-blocks.json`**
  2) Start the framework
  3) Type "`prepare test`"
  4) Edit `test.json`
  5) Type "`make_all test`"
  6) Type "`run_all test`"
  7) Verify the results with the report

- Output :
  ✓ New BB `hello-flood`



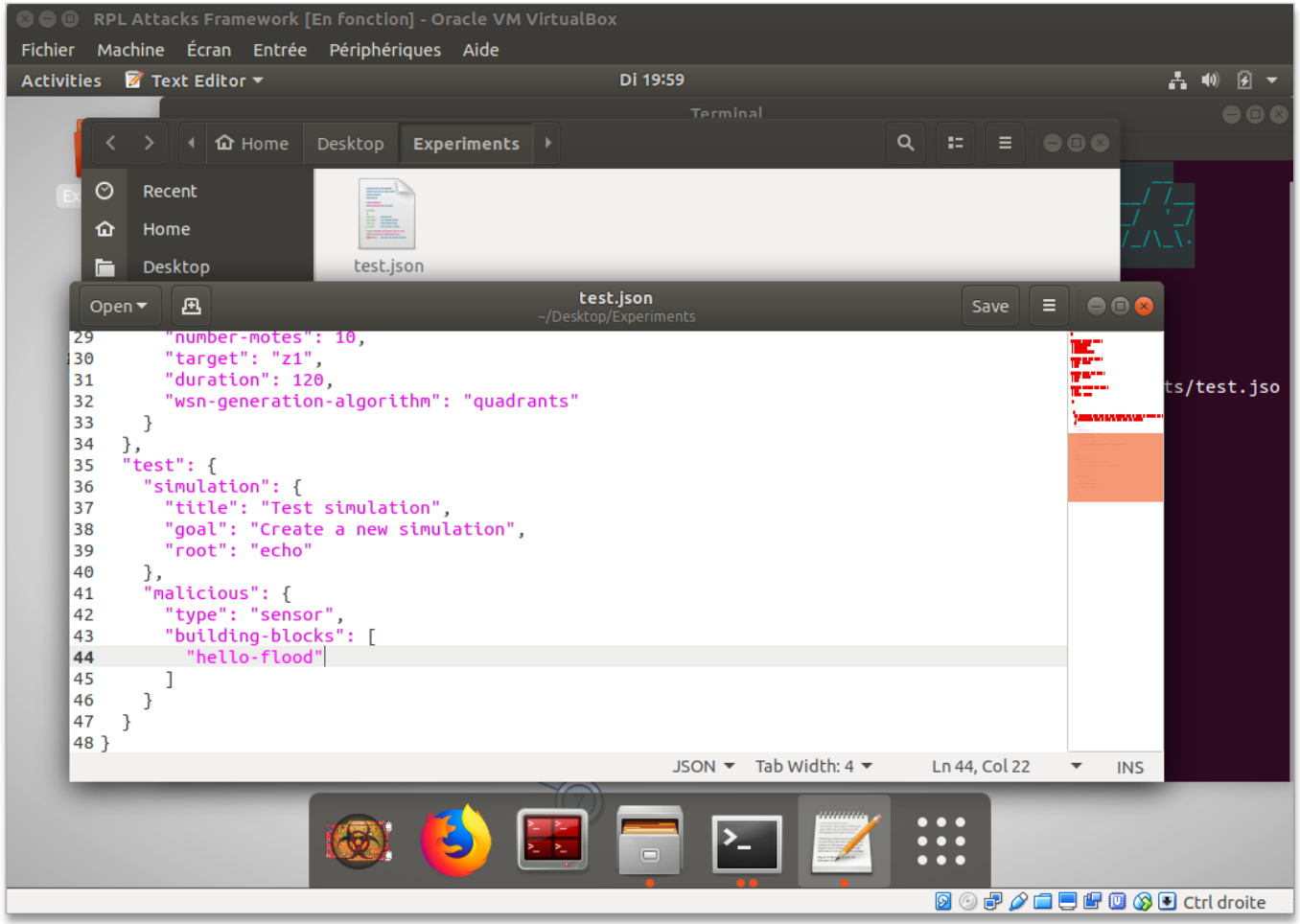
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 7 : Implement *Flooding*

- Start : Vagrant box

- Actions :
  1) Edit `building-blocks.json`
  2) Start the framework
  3) Type "`prepare test`"
  4) **Edit `test.json`**
  5) Type "`make_all test`"
  6) Type "`run_all test`"
  7) Verify the results with the report

- Output :
  ✓ New BB `hello-flood`



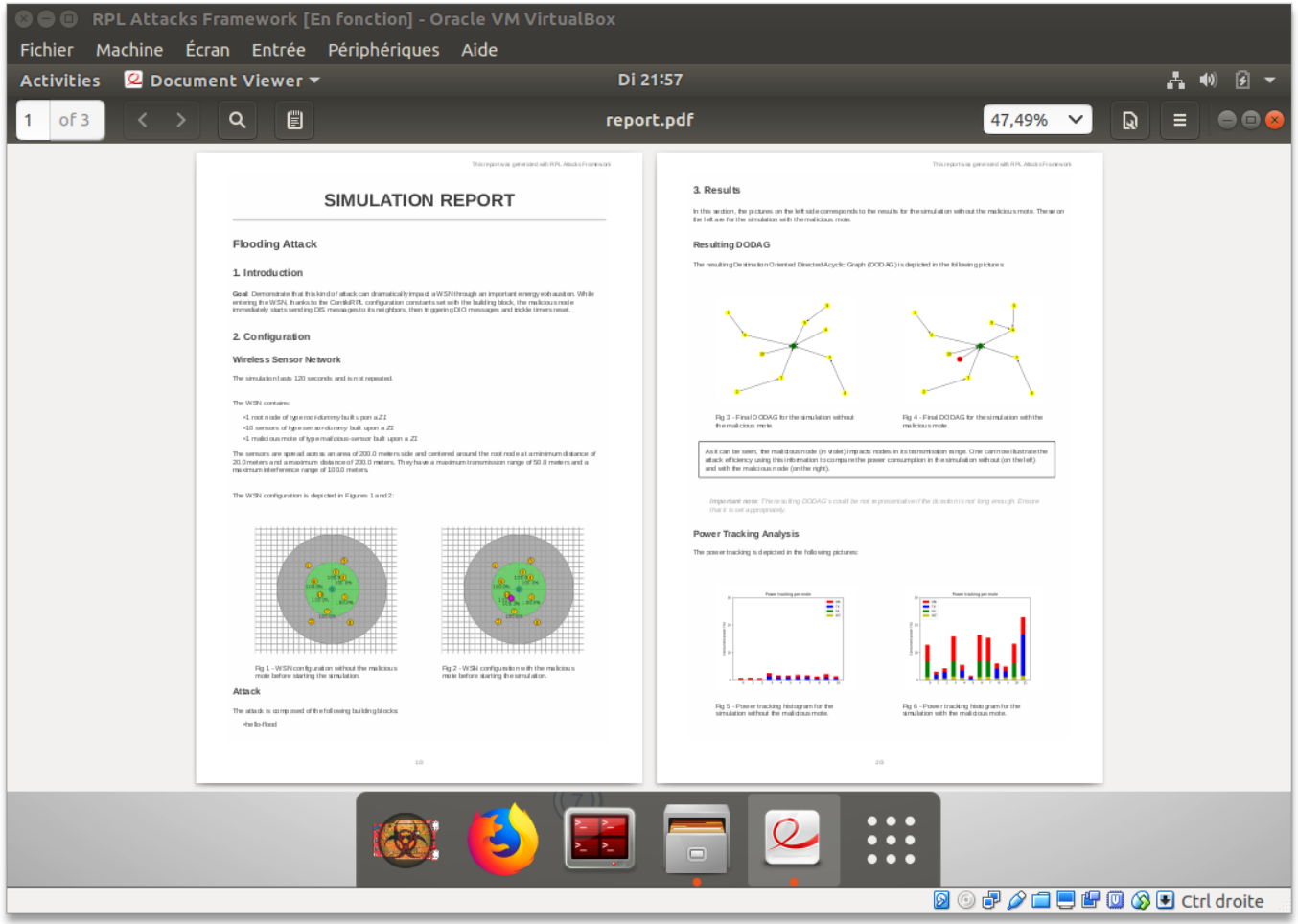
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 7 : Implement *Flooding*

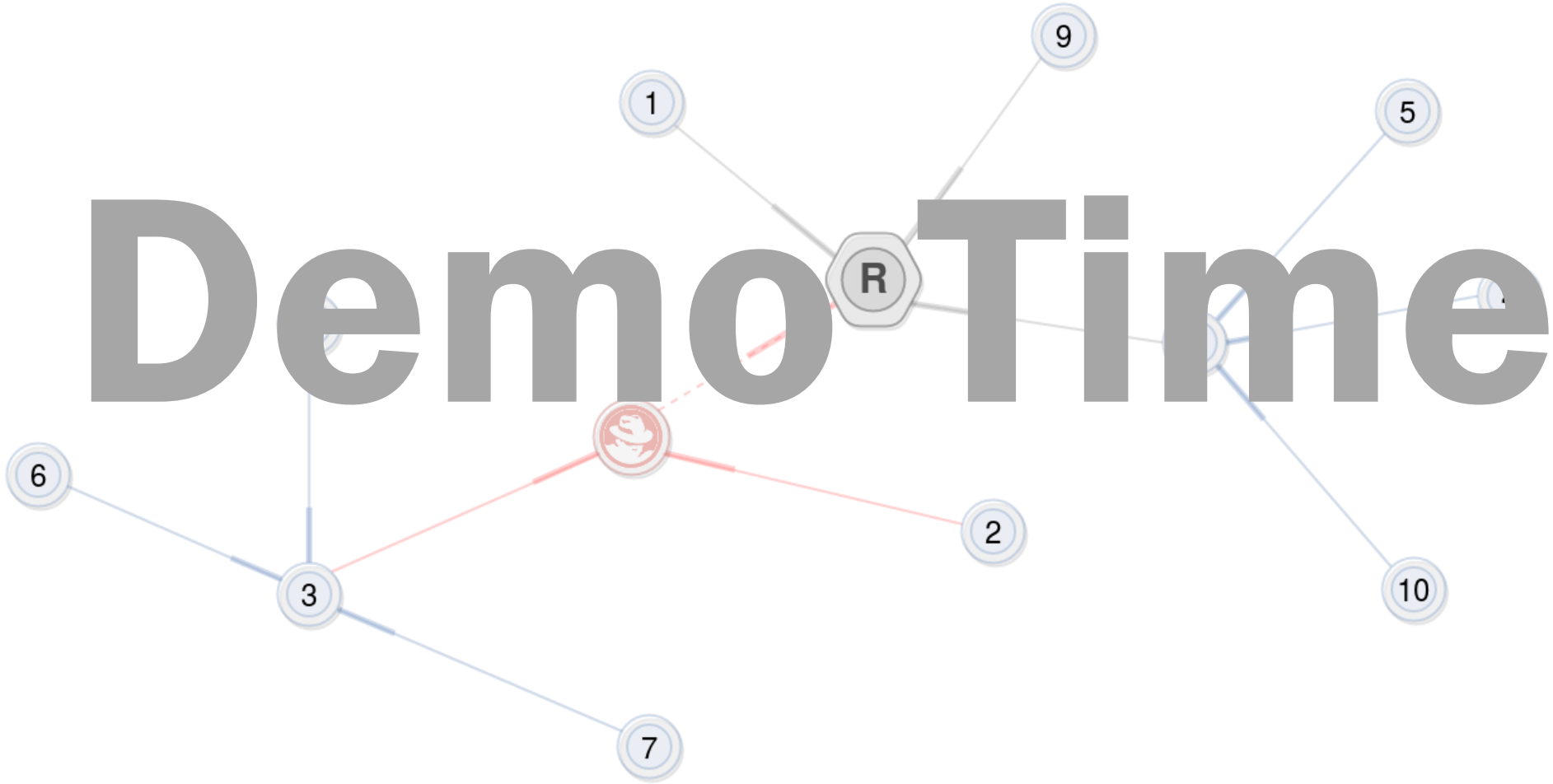
- Start : Vagrant box

- Actions :
  1) Edit `building-blocks.json`
  2) Start the framework
  3) Type "`prepare test`"
  4) Edit `test.json`
  5) Type "`make_all test`"
  6) Type "`run_all test`"
  7) **Verify the results with the report**

- Output :
  ✓ New BB `hello-flood`



Building attacks > Demo 7 : Implement *Flooding*

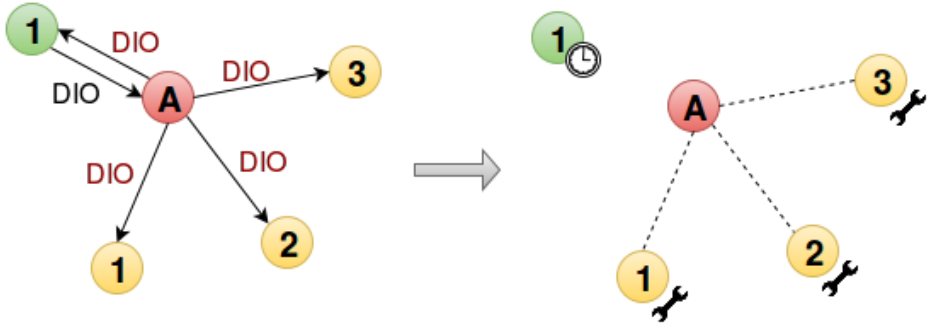


Demo Time

Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 8 : Implement *Version Number*

- Attack :
  - At DIO reception, send a DIO with higher version
  - Causes root to rebuild the DODAG
  - Causes sensors to initiate a global repair



- Decompose :
  - Only 1 BB required : increased version

- Identify :
  - `rpl-icmp6.c` handles DAG version

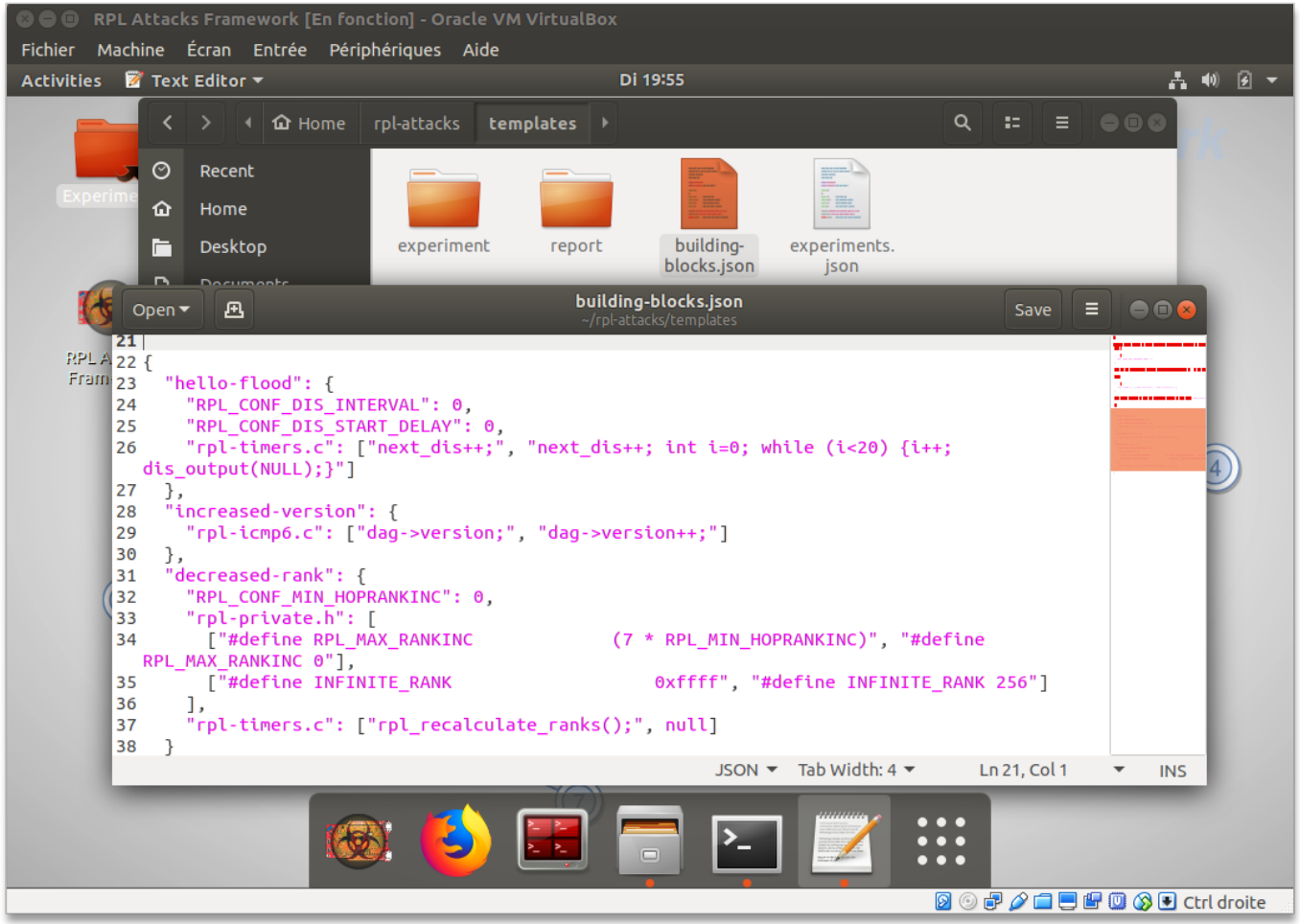
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 8 : Implement *Version Number*

- <u>Start</u> : Vagrant box

- <u>Actions</u> :
  1) **Edit `building-blocks.json`**
  2) Start the framework
  3) Type "`prepare test`"
  4) Edit `test.json`
  5) Type "`make_all test`"
  6) Type "`run_all test`"
  7) Verify the results with the report

- <u>Output</u> :
  ✓ New BB `increased-version`



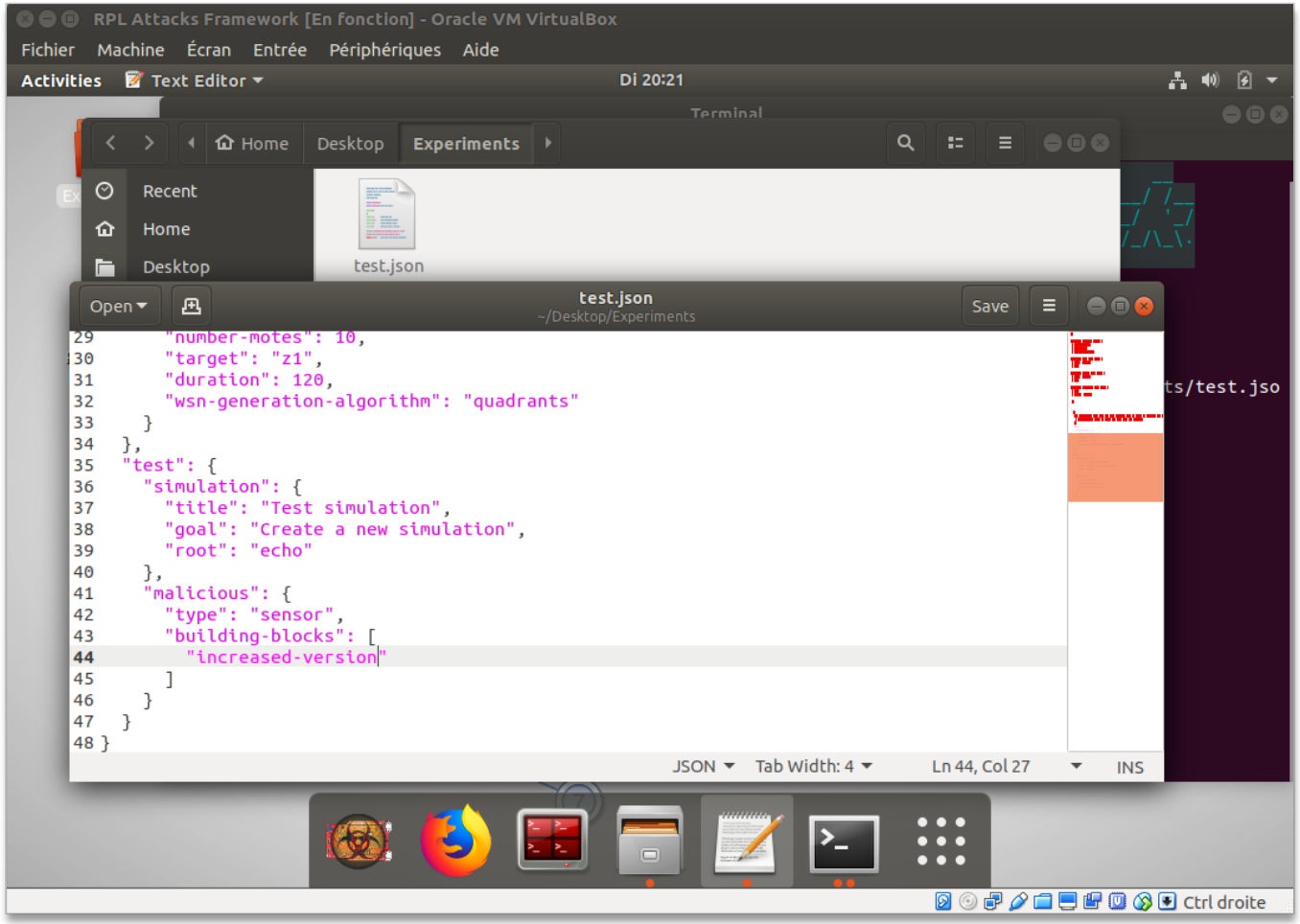
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 8 : Implement *Version Number*

- Start : Vagrant box

- Actions :
  1) Edit `building-blocks.json`
  2) Start the framework
  3) Type "`prepare test`"
  4) **Edit `test.json`**
  5) Type "`make_all test`"
  6) Type "`run_all test`"
  7) Verify the results with the report

- Output :
  ✓ New BB `increased-version`



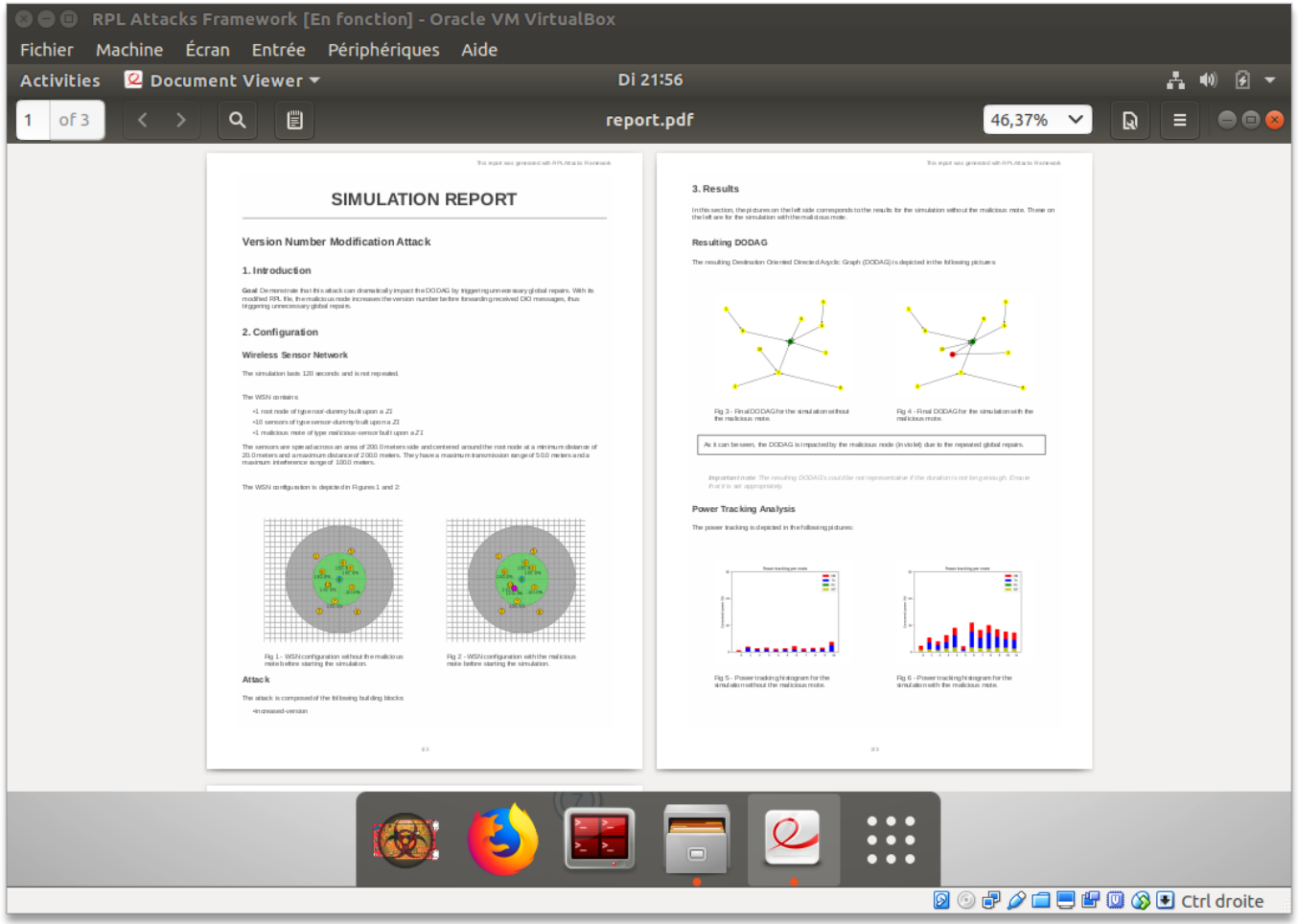
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 8 : Implement *Version Number*

- Start : Vagrant box

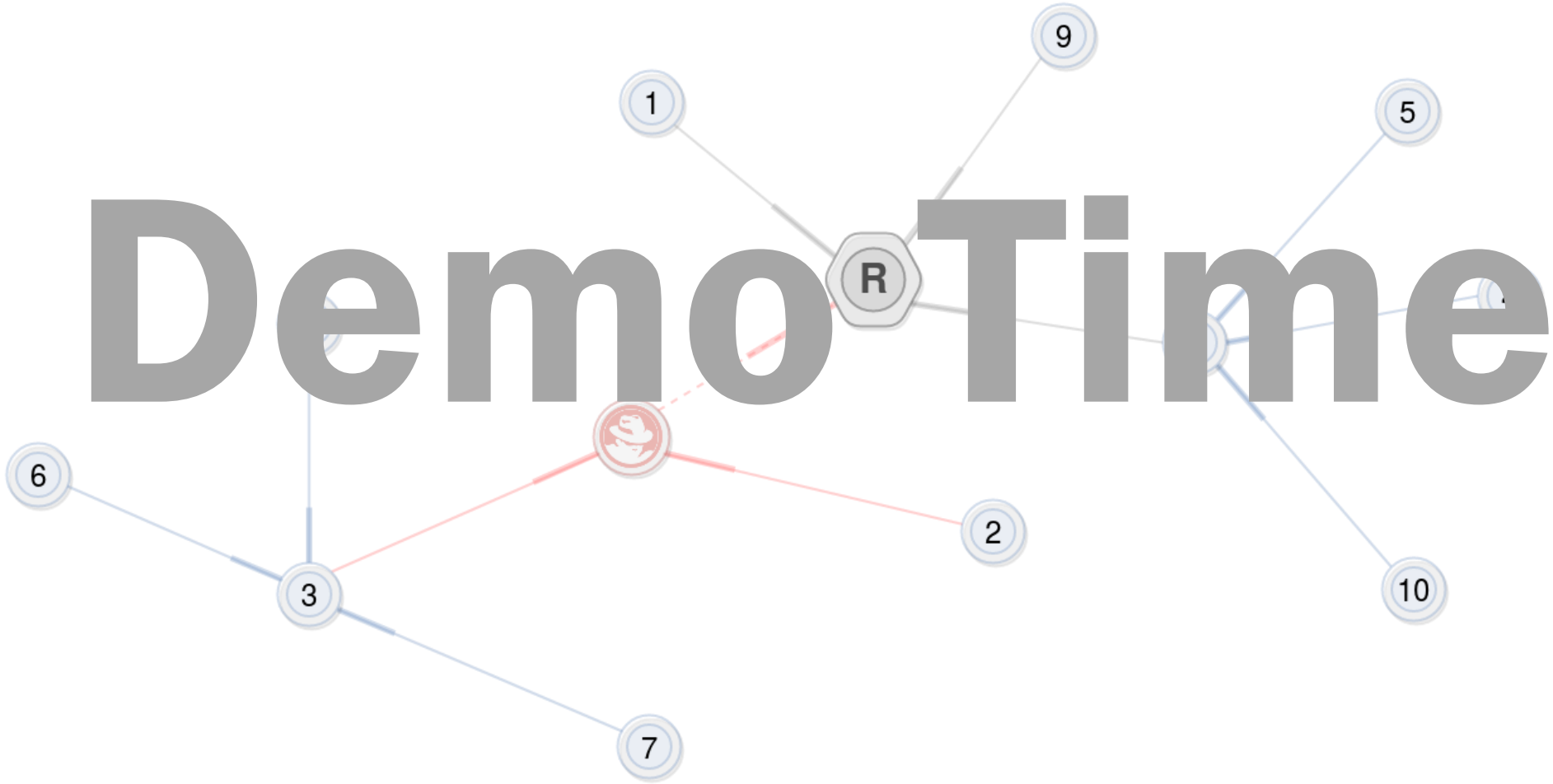
- Actions :
  1) Edit `building-blocks.json`
  2) Start the framework
  3) Type "`prepare test`"
  4) Edit `test.json`
  5) Type "`make_all test`"
  6) Type "`run_all test`"
  7) **Verify the results with the report**

- Output :
  ✓ New BB `increased-version`



Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 8 : Implement *Version Number*

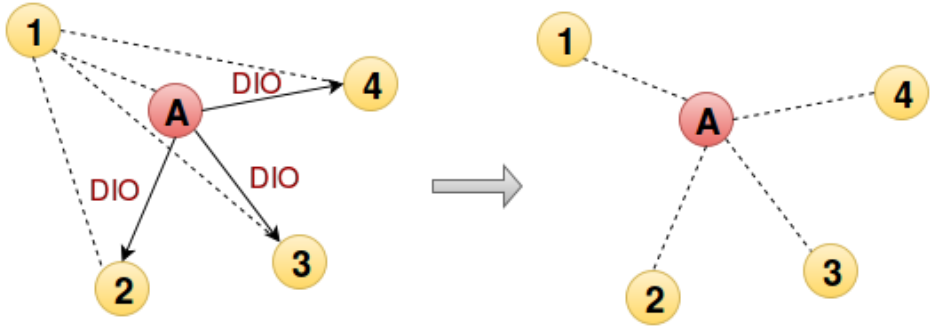


Demo Time

Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 9 : Implement *Decreased Rank*

- Attack :
  - Advertises a DIO with lower rank
  - Causes the malicious mote to become root



- Decompose :
  - Only 1 BB required : decreased rank

- Identify :
  - Depends on RPL constant `RPL_CONF_MIN_HOPRANKINC`
  - `rpl-private.h` contains constants `RPL_MAX_RANKINC` and `INFINITE_RANK`
  - `rpl-timers.c` handles rank recalculation (can be removed)

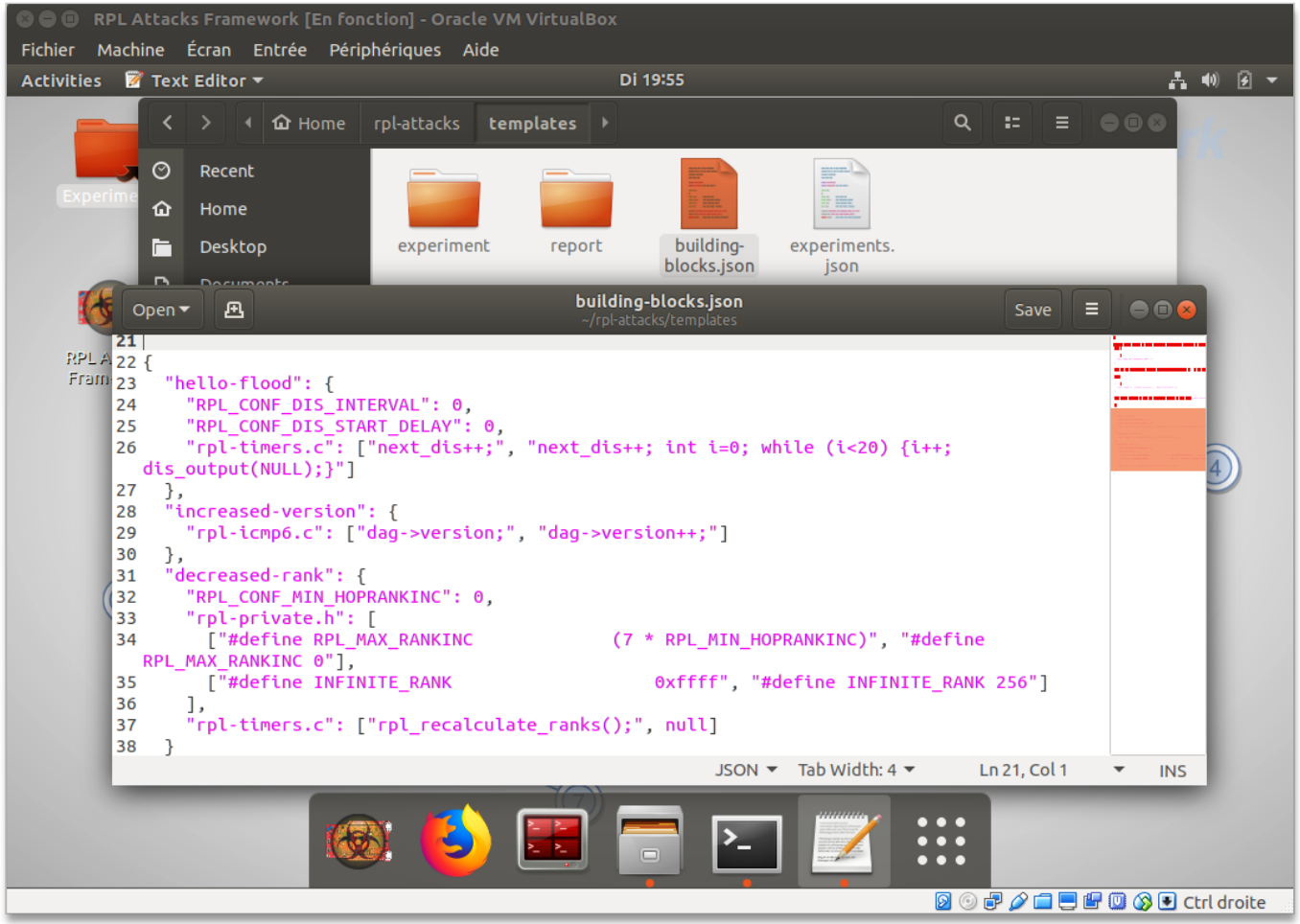
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 9 : Implement *Decreased Rank*

- Start : Vagrant box

- Actions :
  1) **Edit `building-blocks.json`**
  2) Start the framework
  3) Type "`prepare test`"
  4) Edit `test.json`
  5) Type "`make_all test`"
  6) Type "`run_all test`"
  7) Verify the results with the report

- Output :
  ✓ New BB `decreased-rank`



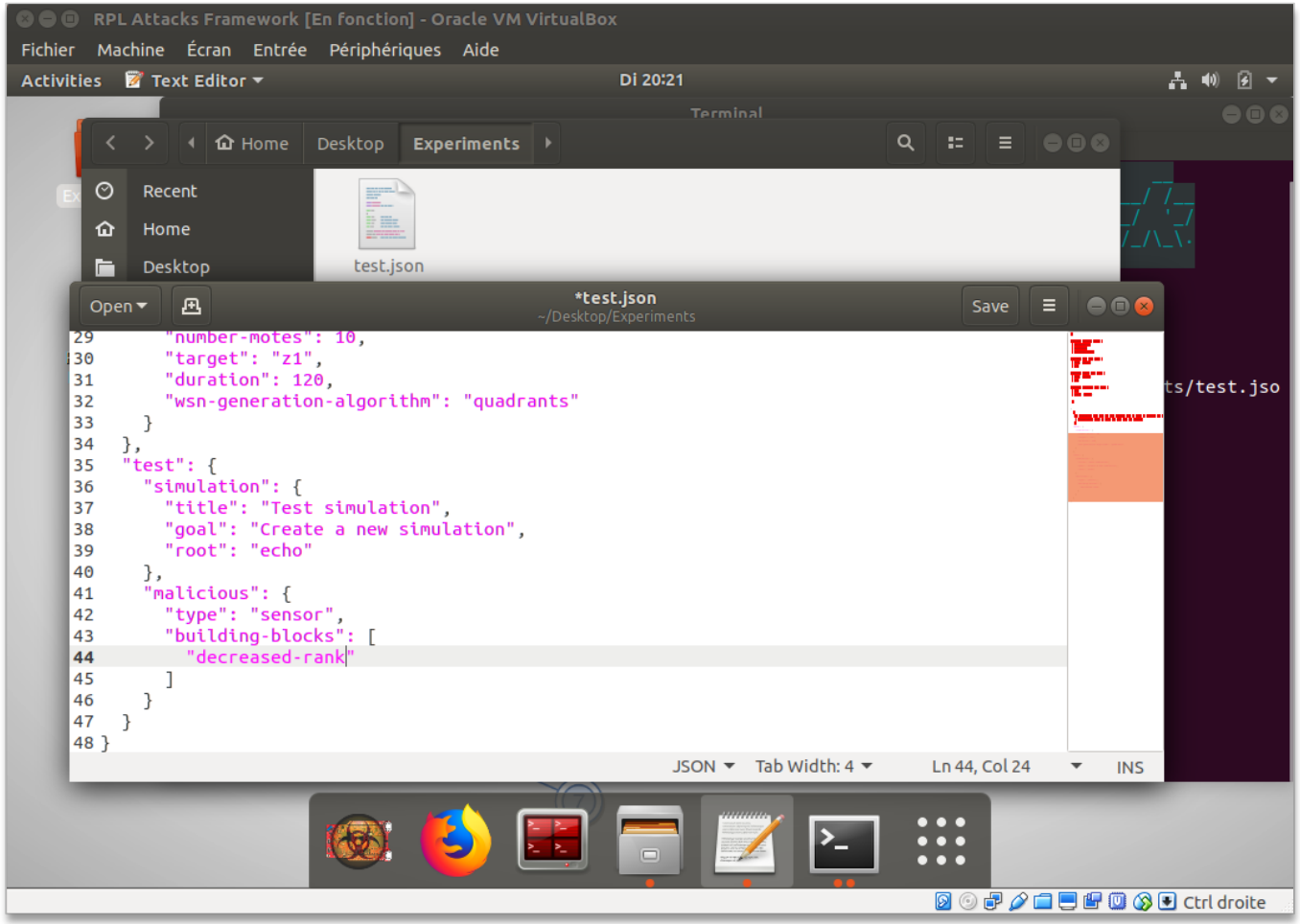
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 9 : Implement *Decreased Rank*

- Start : Vagrant box

- Actions :
  1) Edit `building-blocks.json`
  2) Start the framework
  3) Type "`prepare test`"
  4) **Edit `test.json`**
  5) Type "`make_all test`"
  6) Type "`run_all test`"
  7) Verify the results with the report

- Output :
  ✓ New BB `decreased-rank`



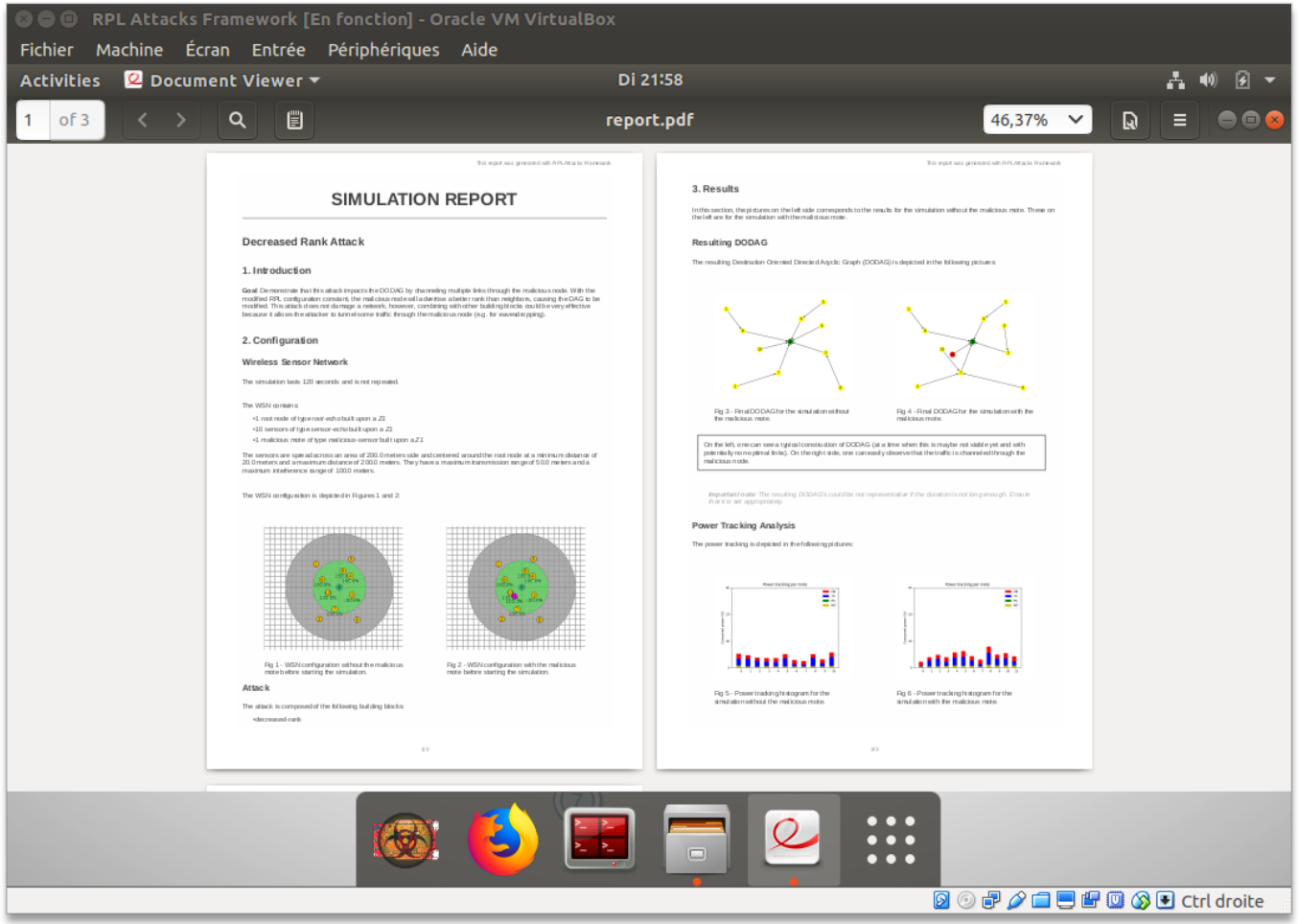
Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 9 : Implement *Decreased Rank*

- Start : Vagrant box

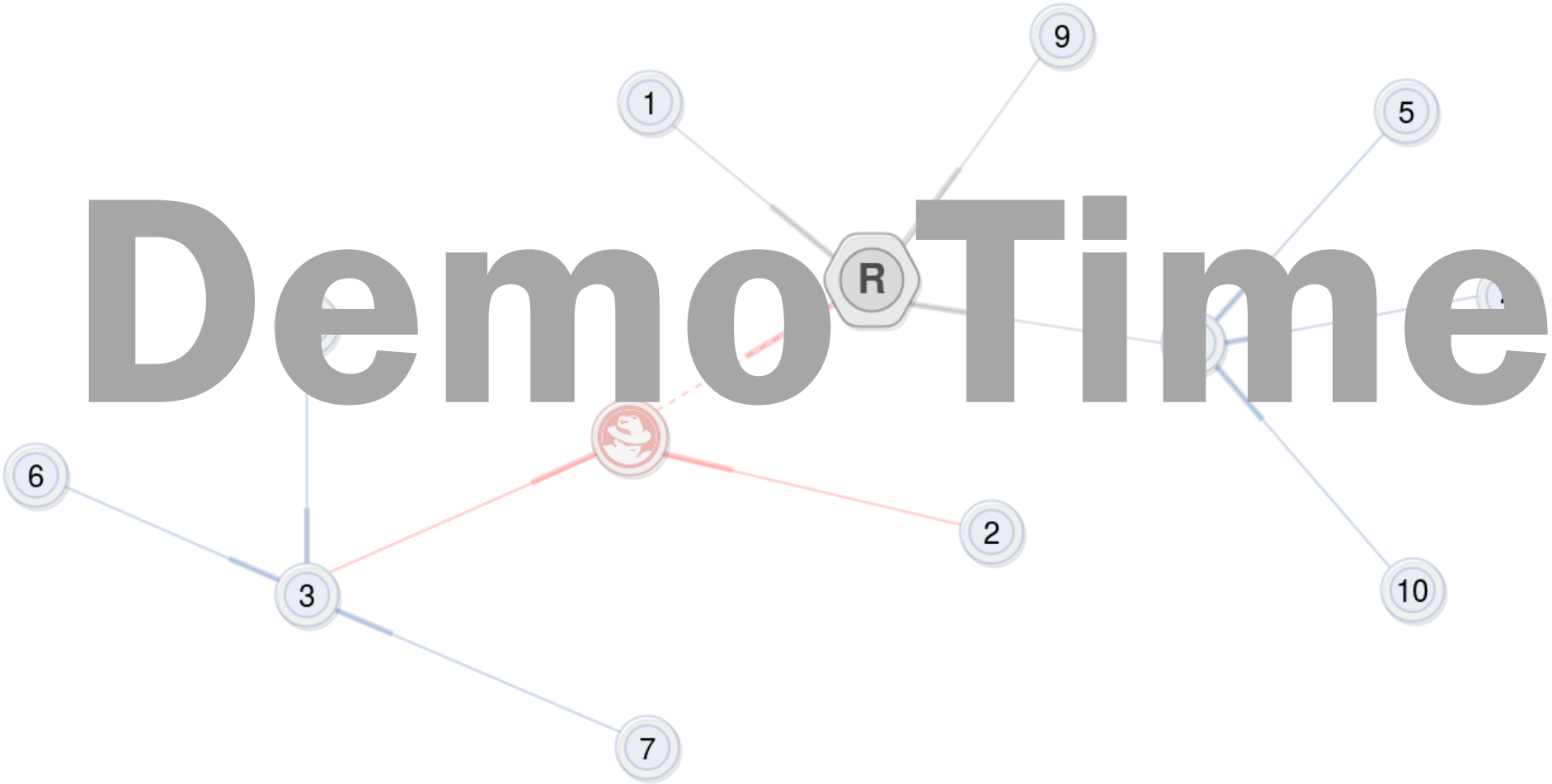
- Actions :
  1) Edit `building-blocks.json`
  2) Start the framework
  3) Type "`prepare test`"
  4) Edit `test.json`
  5) Type "`make_all test`"
  6) Type "`run_all test`"
  7) **Verify the results with the report**

- Output :
  ✓ New BB `decreased-rank`



Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Building attacks > Demo 9 : Implement *Decreased Rank*



Demo Time

Please report any issue at : https://github.com/dhondta/rpl-attacks/issues

Outline

- Introduction
- Background
- RPLA in a nutshell
- Managing simulations
- Building attacks
- Conclusion
  - Objectives
  - Further work

Conclusion > Objectives

1. Different ways to tweak ContikiRPL
2. Easy-to-manage experiments & campaigns
3. Sequential/parallel automation of simulations

   (parallel mode still to be further tested)
4. Easy-to-build malicious sensor

   (not demonstrated)

Conclusion > Further work

- Support multiple malicious sensors per simulation
- Make new building blocks for new attacks
  (cfr *Taxonomy of RPL attacks* [6])
- Integration of Collect View tool (for handling more metrics)
- Add new WSN topology generation algorithms
- Testing on application-level projects

Please submit any enhancement request to : https://github.com/dhondta/rpl-attacks/issues

References

[1] Prof Ramin Sadre, *LINGI2146 – Mobile and Embedded Computing*, UCLouvain

[2] IETF, *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*, at https://tools.ietf.org/html/rfc6550

[3] Texas Instruments Wiki, *Contiki-6LOWPAN*, at http://processors.wiki.ti.com/index.php/Contiki-6LOWPAN

[4] Contiki Community, *Contiki: The Open Source OS for the Internet of Things*, at http://www.contiki-os.org/

[5] Contiki Community, *Get Started with Contiki, About Cooja*, at http://www.contiki-os.org/start.html#start-cooja

[6] A. Mayzaud, R. Badonnel, I. Chrisment, *A Taxonomy of Attacks in RPL based Internet of Things*, International Journal of Network Security, Vol.18, No.3, pp.459-473, May 2016