

# Lab session on IBR-DTN

Ioannis Komnios ([ikomnios@ee.duth.gr](mailto:ikomnios@ee.duth.gr))

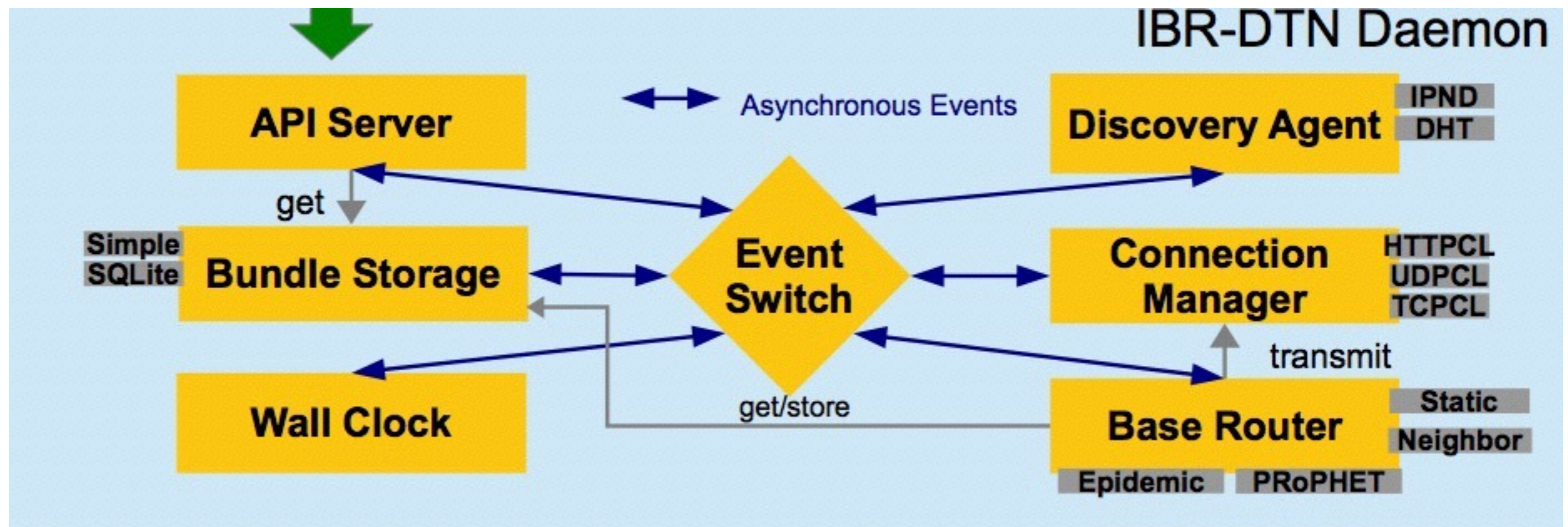


# ④ IBR-DTN overview

---

- ✧ Modular implementation of Bundle Protocol in C++
- ✧ Initially embedded devices and later extended for Android devices
- ✧ Suitable for OpenWRT routers
- ✧ Developed by the Technical University of Braunschweig in 2008 and **still gets updates!**
- ✧ 3 Android apps exist

# 📶 IBR-DTN daemon



# 📶 IBR-DTN documentation

---



❖ Webpage:

<https://www.ibr.cs.tu-bs.de/projects/ibr-dtn/>

❖ Wiki:

<https://trac.ibr.cs.tu-bs.de/project-cm-2012-ibrdtn/wiki>

# Ⓢ Supported systems

---



✧ Installation available for:

✧ OpenWRT

✧ **Debian/Ubuntu**

✧ Debian ARM

✧ MacOS X

✧ Gentoo Linux








✧ Windows



# ① Finding the right one

- ✦ Check the following repository list to find the right distribution for your system:

[http://download.opensuse.org/repositories/home:/j\\_morgenroth/](http://download.opensuse.org/repositories/home:/j_morgenroth/)

Index of /repositories/home:/j_morgenroth		
Name	Last modified	Size
 Parent Directory		-
 Debian_7.0/	11-May-2015 18:45	-
 Debian_8.0/	11-May-2015 18:49	-
 xUbuntu_12.04/	11-May-2015 18:44	-
 xUbuntu_14.04/	11-May-2015 18:47	-
 xUbuntu_14.10/	11-May-2015 18:55	-
 xUbuntu_15.04/	11-May-2015 18:45	-
Apache/2.2.12 (Linux/SUSE) Server at download.opensuse.org Port 80		
MirrorBrain powered by Apache		

# ☎ ...and authenticating it



- ❖ First, we need to add the corresponding PGP key to our apt keyring in order to be able to authenticate the package:

```
wget -O - http://download.opensuse.org/repositories/home:/j_morgenroth/[distribution]/Release.key | \sudo apt-key add -
```

- ❖ In our case:

```
wget -O - http://download.opensuse.org/repositories/home:/j_morgenroth/xUbuntu_14.04/Release.key | \sudo apt-key add -
```

```
ikomnlos@ikomnlos:/etc/apt$ wget -O - http://download.opensuse.org/repositories/home:/j_morgenroth/xUbuntu_14.04/Release.key | \sudo apt-key add -
--2016-03-11 22:43:32-- http://download.opensuse.org/repositories/home:/j_morgenroth/xUbuntu_14.04/Release.key
Resolving download.opensuse.org (download.opensuse.org)... 2001:67c:2178:8::13, 195.135.221.134
Connecting to download.opensuse.org (download.opensuse.org)|2001:67c:2178:8::13|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1019 [application/pgp-keys]
Saving to: 'STDOUT'

100%[=====>] 1019 --.-K/s in 0s

2016-03-11 22:43:32 (109 MB/s) - written to stdout [1019/1019]

OK
```

# ☉ Update repositories list

---



✧ Then, we find the list of repositories (/etc/apt/sources.list)

and add the corresponding URL:

```
deb http://download.opensuse.org/repositories/  
home:/j_morgenroth/[distribution] ./
```

✧ In our case:

```
deb http://download.opensuse.org/repositories/  
home:/j_morgenroth/xUbuntu_14.04 ./
```

# 📶 Update and install

---



```
sudo apt-get update
```

```
sudo apt-get install ibrdtn ibrdtn-tools
```

# ☉ Alternately

---



✧ Install from source

✧ Download the source code:

<https://trac.ibr.cs.tu-bs.de/project-cm-2012-ibrdtn/wiki/source>

✧ Unzip (tar), configure (./configure), build (make) and copy the executables to the right directories (make install)

ibrcommon-1.0.1.tar.gz

ibrdtn-1.0.1.tar.gz

ibrdtnd-1.0.1.tar.gz

ibrdtn-tools-1.0.1.tar.gz

# ⊙ Configuration file

---



- ❖ A sample configuration file is available in  
`/etc/ibrdtn/ibrdtnd.conf`
- ❖ You can either run the default configuration file  
OR  
Copy this sample configuration file to another  
folder (e.g. Desktop) and make your changes

# Ⓢ Configuration file



```
1 #####
2 # IBR-DTN daemon #
3 #####
4
5 #
6 # the local eid of the dtn node
7 # default is the hostname
8 #
9 #local_uri = dtn://node.dtn
10
11 #
12 # specifies an additional logfile
13 #
14 logfile = /var/log/ibrdtn/ibrdtn.log
15
16 #
17 # Limit the block size of all bundles.
18 #
19 # The value accepts different multipliers.
20 # G = 1,000,000,000 bytes
21 # M = 1,000,000 bytes
22 # K = 1,000 bytes
23 #
24 #limit_blocksize = 1.3G
25
```

```
26 #
27 # Limit the block size of foreign bundles.
28 # Foreign bundles are not address from or to the
29 # local node.
30 #
31 # The value accepts different multipliers.
32 # G = 1,000,000,000 bytes
33 # M = 1,000,000 bytes
34 # K = 1,000 bytes
35 #
36 #limit_foreign_blocksize = 500M
```

# Ⓢ Configuration file

---



```
44 #
45 # Limit the max. lifetime of a bundle.
46 # Bundles with a lifetime greater than this value will be rejected.
47 #
48 #limit_lifetime = 604800
49
50 # limit the numbers of bundles in transit (default: 5)
51 #limit_bundles_in_transit = 5
52
```

```
62 #
63 # enable fragmentation support
64 # (default is enabled)
65 #
66 #fragmentation = no
67
68 #
69 # if fragmentation is enabled, it is possible to split up
70 # bundles larger than a specific limit into fragments
71 #
72 # limit_payload = 500K
73
```

# ① Configuration file

---



```
115 #####
116 # convergence layer configuration #
117 #####
118
119 #
120 # discovery over UDP/IP
121 #
122 # You can specify an multicast address to listen to for discovery announcements.
123 # If no address is specified the multicast equivalent of broadcast is used.
124 #
125 discovery_address = ff02::142 224.0.0.142
126
127 # Specify how often discovery beacons are sent. The default is every 5 seconds.
128 #discovery_interval = 5
129
```



```

162 #
163 # Defines the interface with global internet access. With this definition
164 # the daemon can detect internet access by its own and might assume specific
165 # nodes as available or unavailable depending on the internet state.
166 #
167 #net_internet = eth0
168
169 #
170 # configuration for a convergence layer named lan0
171 #
172 #net_lan0_type = tcp # we want to use TCP as protocol
173 #net_lan0_interface = eth0 # listen on interface eth0
174 #net_lan0_port = 4556 # with port 4556 (default)
175
176 #
177 # configuration for a convergence layer named lan1
178 #
179 #net_lan1_type = udp # we want to use UDP as protocol
180 #net_lan1_interface = eth0 # listen on interface eth0
181 #net_lan1_port = 4556 # with port 4556 (default)
182

```

# ① Configuration file



```
213 #####
214 # routing configuration          #
215 #####
216
217 #
218 # routing strategy
219 #
220 # values: default | epidemic | flooding | prophet | none
221 #
222 # In the "default" the daemon only delivers bundles to neighbors and static
223 # available nodes. The alternative module "epidemic" spread all bundles to
224 # all available neighbors. Flooding works like epidemic, but do not send the
225 # own summary vector to neighbors. Prophet forwards based on the probability
226 # to encounter other nodes (see RFC 6693).
227 #
228 routing = prophet
229
230 #
231 # forward bundles to other nodes (yes/no)
232 #
233 #routing_forwarding = yes
234 ..
```

# ① Configuration file

---



```
314 #####
315 # bundle security protocol      #
316 #####
317
318 #
319 # the level specifies the security constrains
320 #
321 # 0 = no constrains (default)
322 # 1 = accept only authenticated bundles
323 # 2 = accept only encrypted bundles
324 # 4 = accept only signed bundles
325 #
326 # Combination is allowed by adding values
327 # e.g. 5 = accept only bundles which are signed AND authenticated
328 #
329 #security_level = 0
330
```

# Available interfaces?



ifconfig

```
ikomnios@ikomnios:/etc/apt$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:28:d4:6f
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe28:d46f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1225 errors:0 dropped:0 overruns:0 frame:0
          TX packets:852 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:836266 (836.2 KB)  TX bytes:79414 (79.4 KB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:e4:d3:2d
          inet addr:192.168.1.4  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: 2a02:587:1813:8900:a00:27ff:fee4:d32d/64 Scope:Global
          inet6 addr: 2a02:587:1813:8900:c96c:87cb:f91a:cd5a/64 Scope:Global
          inet6 addr: fe80::a00:27ff:fee4:d32d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4227 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2101 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4981827 (4.9 MB)  TX bytes:245775 (245.7 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:222 errors:0 dropped:0 overruns:0 frame:0
          TX packets:222 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:27167 (27.1 KB)  TX bytes:27167 (27.1 KB)
```

# ☉ Daemon options

---



```
^Cikomnios@ikomnios:/etc/apt$ dtnd --help
IBR-DTN version: 1.0.1 (build 294b543)
Syntax: dtnd [options]
-h|--help      display this text
-c <file>      set a configuration file
-D            daemonize the process
-k            stop the running daemon
-p <file>      store the pid in this pidfile
-i <interface> interface to bind on (e.g. eth0)
-d <level>     enable debugging and set a verbose level
-q            enables the quiet mode (no logging to the console)
-t <threads>   specify a number of threads for parallel event processing
-v            be verbose - show NOTICE log messages
--version      show version and exit
--noapi        disable API module
--nodiscovery  disable discovery module
--timestamp    enables timestamps for logging instead of datetime values
ikomnios@ikomnios:/etc/apt$
```

# ☉ Running the daemon

---



`dtnd -i interface_name`

In this case: `dtnd -i eth1`

*\*This will run the default configuration file\**

```
ikomnios@ikomnios:/etc/apt$ dtnd -i eth1
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: IBR-DTN daemon 1.0.1 (build 294b543)
Fri Mar 11 22:46:33 2016 INFO Configuration: Using default settings. Call with --help for options.
Fri Mar 11 22:46:33 2016 INFO BundleCore: Local node name: dtn://ikomnios
Fri Mar 11 22:46:33 2016 INFO BundleCore: Forwarding of bundles enabled.
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: using bundle storage in memory-only mode
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: API initialized using tcp socket: loopback:4550
Fri Mar 11 22:46:33 2016 ERROR ApiServer: Cannot bind to socket with address [::1]:4550
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: TCP ConvergenceLayer added on eth1:4556
Fri Mar 11 22:46:33 2016 INFO DiscoveryAgent: listen to [ff02::142]:4551
Fri Mar 11 22:46:33 2016 INFO DiscoveryAgent: listen to [224.0.0.142]:4551
Fri Mar 11 22:46:33 2016 INFO DiscoveryAgent: add interface eth1
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: Using default routing extensions
```

# 📶 Running the daemon



`dtnd -i interface_name`

In this case: `dtnd -i eth1`

Did anyone  
notice this?

```
ikomnios@ikomnios:/etc/apt$ dtnd -i eth1
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: IBR-DTN daemon 1.0.1 (build 294b5f3)
Fri Mar 11 22:46:33 2016 INFO Configuration: Using default settings. Call with --help for options.
Fri Mar 11 22:46:33 2016 INFO BundleCore: Local node name: dtn://ikomnios
Fri Mar 11 22:46:33 2016 INFO BundleCore: Forwarding of bundles enabled.
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: using bundle storage in memory-only mode
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: APT initialized using tcp socket: loopback:4550
Fri Mar 11 22:46:33 2016 ERROR ApiServer: Cannot bind to socket with address [::1]:4550
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: TCP ConvergenceLayer added on eth1:4556
Fri Mar 11 22:46:33 2016 INFO DiscoveryAgent: listen to [ff02::142]:4551
Fri Mar 11 22:46:33 2016 INFO DiscoveryAgent: listen to [224.0.0.142]:4551
Fri Mar 11 22:46:33 2016 INFO DiscoveryAgent: add interface eth1
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: Using default routing extensions
```

# 📶 Running the daemon



`dtnd -i interface_name`

In this case: `dtnd -i eth1`

Another  
instance of *dtnd* is  
running on the  
background

```
ikomnios@ikomnios:/etc/apt$ dtnd -i eth1
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: IBR-DTN daemon 1.0.1
Fri Mar 11 22:46:33 2016 INFO Configuration: Using default settings for options.
Fri Mar 11 22:46:33 2016 INFO BundleCore: Local node name: dtn://...
Fri Mar 11 22:46:33 2016 INFO BundleCore: Forwarding of bundles enabled.
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: using bundle storage in memory-only mode
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: APT initialized using tcp socket: loopback:4550
Fri Mar 11 22:46:33 2016 ERROR ApiServer: Cannot bind to socket with address [::1]:4550
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: TCP ConvergenceLayer added on eth1:4556
Fri Mar 11 22:46:33 2016 INFO DiscoveryAgent: listen to [ff02::142]:4551
Fri Mar 11 22:46:33 2016 INFO DiscoveryAgent: listen to [224.0.0.142]:4551
Fri Mar 11 22:46:33 2016 INFO DiscoveryAgent: add interface eth1
Fri Mar 11 22:46:33 2016 INFO NativeDaemon: Using default routing extensions
```



# ☉ Stopping dtnd

---

`sudo service ibrdtn stop`

```
^Cikomnios@ikomnios:/etc/apt$ sudo service ibrdtn stop
[sudo] password for ikomnios:
ikomnios@ikomnios:/etc/apt$ dtnd -i eth1
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: IBR-DTN daemon 1.0.1 (build 294b543)
Fri Mar 11 23:49:13 2016 INFO Configuration: Using default settings. Call with --help for options.
Fri Mar 11 23:49:13 2016 INFO BundleCore: Local node name: dtn://ikomnios
Fri Mar 11 23:49:13 2016 INFO BundleCore: Forwarding of bundles enabled.
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: using bundle storage in memory-only mode
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: API initialized using tcp socket: loopback:4550
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: TCP ConvergenceLayer added on eth1:4556
Fri Mar 11 23:49:13 2016 INFO DiscoveryAgent: listen to [ff02::142]:4551
Fri Mar 11 23:49:13 2016 INFO DiscoveryAgent: listen to [224.0.0.142]:4551
Fri Mar 11 23:49:13 2016 INFO DiscoveryAgent: add interface eth1
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: Using default routing extensions
```

# ☎ Stopping dtnd



```
sudo service ibrdtn stop
```

Solved!

```
^Cikomnios@ikomnios:/etc/apt$ sudo service ibrdtn stop
[sudo] password for ikomnios:
ikomnios@ikomnios:/etc/apt$ dtnd -i eth1
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: IBR-DTN daemon 1.0.1 (build 294b543)
Fri Mar 11 23:49:13 2016 INFO Configuration: Using default settings. Call with --help for options.
Fri Mar 11 23:49:13 2016 INFO BundleCore: Local node name: dtn://ikomnios
Fri Mar 11 23:49:13 2016 INFO BundleCore: Forwarding of bundles enabled.
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: using bundle storage in memory-only mode
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: API initialized using tcp socket: loopback:4550
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: TCP ConvergenceLayer added on eth1:4556
Fri Mar 11 23:49:13 2016 INFO DiscoveryAgent: listen to [ff02::142]:4551
Fri Mar 11 23:49:13 2016 INFO DiscoveryAgent: listen to [224.0.0.142]:4551
Fri Mar 11 23:49:13 2016 INFO DiscoveryAgent: add interface eth1
Fri Mar 11 23:49:13 2016 INFO NativeDaemon: Using default routing extensions
```

# 📶 Your own configuration file

---



If you want to run you own configuration file:

- Enter the folder where the configuration file is stored
- Execute dtnd as follows:

```
dtnd -i eth1 -c myConfigurationFile.conf
```

# 📶 dtnping



In a new terminal tab: `dtnping nodename/echo`

In this case: `dtnping dtn://ikomnios/echo`

```
ikomnios@ikomnios:/etc/apt$ dtnping dtn://ikomnios/echo
ECHO dtn://ikomnios/echo 64 bytes of data.
64 bytes from dtn://ikomnios/echo: seq=1 ttl=30 time=37.20 ms
64 bytes from dtn://ikomnios/echo: seq=2 ttl=30 time=1.33 ms
64 bytes from dtn://ikomnios/echo: seq=3 ttl=30 time=1.47 ms
64 bytes from dtn://ikomnios/echo: seq=4 ttl=30 time=0.94 ms
64 bytes from dtn://ikomnios/echo: seq=5 ttl=30 time=2.02 ms
64 bytes from dtn://ikomnios/echo: seq=6 ttl=30 time=1.67 ms
64 bytes from dtn://ikomnios/echo: seq=7 ttl=30 time=1.25 ms
64 bytes from dtn://ikomnios/echo: seq=8 ttl=30 time=2.38 ms
64 bytes from dtn://ikomnios/echo: seq=9 ttl=30 time=1.69 ms
64 bytes from dtn://ikomnios/echo: seq=10 ttl=30 time=1.87 ms
^C
--- dtn://ikomnios/echo echo statistics ---
10 bundles transmitted, 10 received, 0.00% bundle loss, time 9.81 s
rtt min/avg/max = 0.94/5.18/37.20 ms
ikomnios@ikomnios:/etc/apt$
```

In a similar way we can also ping another host. Give it a try!

# 📶 dtnsend - dtnrecv

---



- ❖ Two different hosts run IBRDTN:
  - ❖ Host 1: `dtn:/ /ikomnios`
  - ❖ Host 2: `dtn:/ /ikomnios-VirtualBox`
- ❖ Host 1 creates a file (that includes a message) to send to Host 2
  - ❖ `echo This file is for Host 2! > myFile`

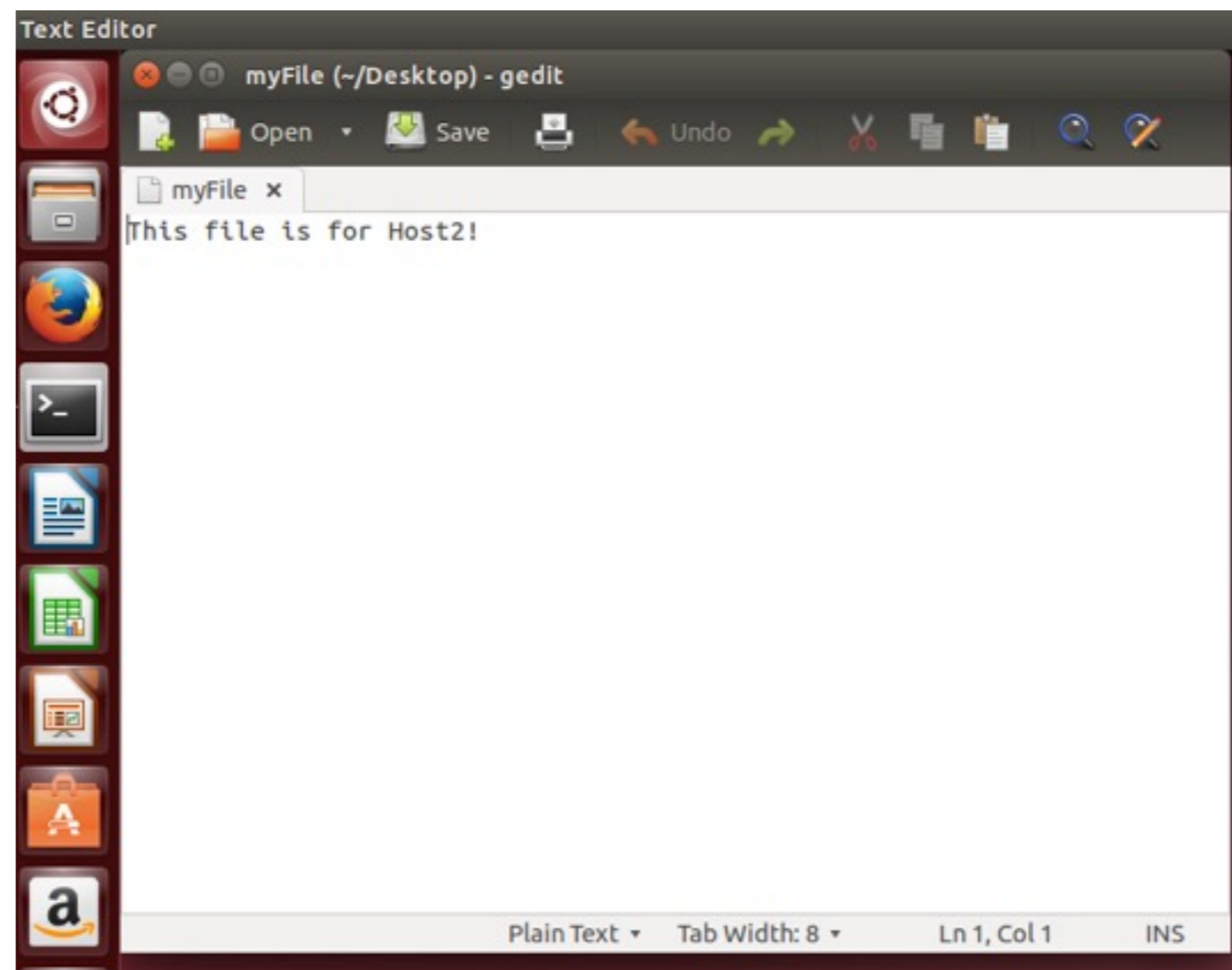
Was the file created?

# ☉ Host 1 Terminal & Desktop

---



```
ikomnios@ikomnios:~/Desktop$ echo This file is for Host2! > myFile  
ikomnios@ikomnios:~/Desktop$
```



# 📶 dtnsend - dtnrecv

---



✦ Host 2 sets up a dtnReceiver to receive the message

✦ `dtnrecv --name dtnReceiver`

```
ikomnios@ikomnios-VirtualBox:~/Desktop$ dtnrecv --name dtnReceiver
```

✦ Host 1 send the file to Host 2

✦ `dtnsend dtn://ikomnios-VirtualBox/dtnReceiver myFile`

```
ikomnios@ikomnios:~/Desktop$ dtnsend dtn://ikomnios-VirtualBox/dtnReceiver myFile
Transfer file "myFile" to dtn://ikomnios-VirtualBox/dtnReceiver
ikomnios@ikomnios:~/Desktop$
```

Was the file sent?

# ① dtnsend - dtnrecv

---



✧ Host 2 sets up a dtnReceiver to receive the message

✧ `dtnrecv --name dtnReceiver`

```
ikomnios@ikomnios-VirtualBox:~/Desktop$ dtnrecv --name dtnReceiver
```

✧ Host 1 send the file to Host 2

✧ `dtnsend dtn://ikomnios-VirtualBox/dtnReceiver myFile`

```
ikomnios@ikomnios:~/Desktop$ dtnsend dtn://ikomnios-VirtualBox/dtnReceiver myFile
Transfer file "myFile" to dtn://ikomnios-VirtualBox/dtnReceiver
ikomnios@ikomnios:~/Desktop$
```

```
ikomnios@ikomnios-VirtualBox:~/Desktop$ dtnrecv --name dtnReceiver
This file is for Host2!
ikomnios@ikomnios-VirtualBox:~/Desktop$
```

# ① dtnsend - dtnrecv

---



Is the file  
sent?

✦ Now disable the interface of Host 2 and let Host 1 send one more file

# 📶 dtnsend - dtnrecv

---



- ❖ Now disable the interface of Host 2 and let Host 1 send one more file
- ❖ Re-enable the interface.

What  
happens?

# 📶 dtninbox - dtnoutbox

---



- ❖ The tools `dtninbox` and `dtnoutbox` can automatize the process of sending and receiving bundles
- ❖ If a new file is created in the outbox, it will be delivered into the inbox automatically.

# 📶 dtninbox - dtnoutbox

---



- ❖ Host 1 (dtn: / / ikomnios) is the receiving side  
dtninbox inboxReceiver InboxFolderName/

```
ikomnios@ikomnios:~/Desktop$ mkdir inboxFolder  
ikomnios@ikomnios:~/Desktop$ dtninbox inboxReceiver inboxFolder/
```

- ❖ Host 2 (dtn: / / ikomnios-VirtualBox) is the sending side  
dtnoutbox outboxSender OutboxFolderName/ Host1Name/  
inboxReceiver

```
ikomnios@ikomnios-VirtualBox:~/Desktop$ mkdir outboxFolder  
ikomnios@ikomnios-VirtualBox:~/Desktop$ dtnoutbox outboxSender outboxFolder/ dtn://ikomnios/inboxReceiver  
-- dtnoutbox --
```

# 📶 dtninbox - dtnoutbox

---



Now add a file in the outboxFolder? What happens?

# 📶 dtninbox - dtnoutbox

---



```
ikomnios@ikomnios-VirtualBox:~/Desktop$ mkdir outboxFolder
ikomnios@ikomnios-VirtualBox:~/Desktop$ dtnoutbox outboxSender outboxFolder/ dtn://ikomnios/inboxReceiver
-- dtnoutbox --
file found: sharedDocument
files sent: sharedDocument
```

Now add a file in the outboxFolder? What happens?

```
ikomnios@ikomnios:~/Desktop$ mkdir inboxFolder
ikomnios@ikomnios:~/Desktop$ dtninbox inboxReceiver inboxFolder/
received bundle: [511054446.1] dtn://ikomnios-VirtualBox/outboxSender
```

# Recap

---



- ❖ IBR-DTN installation in Ubuntu 14.04

- ❖ Default configuration file

- ❖ dtnd  
dtnping  
dtnsend  
dtnrecv  
dtninbox  
dtnoutbox

# 📶 IBR-DTN apps

---

- ✦ Also check out the IBR-DTN version available for Android mobile devices in Google Play Store:
- ✦ IBR-DTN (the IBR-DTN daemon for Android)
- ✦ ShareBox (shared folder application)
- ✦ Whisper (messaging application)
- ✦ Talkie (walkie-talkie application)



Thank you for  
your attention!

---

Any questions?