ns-3 Training

Direct Code Execution overview

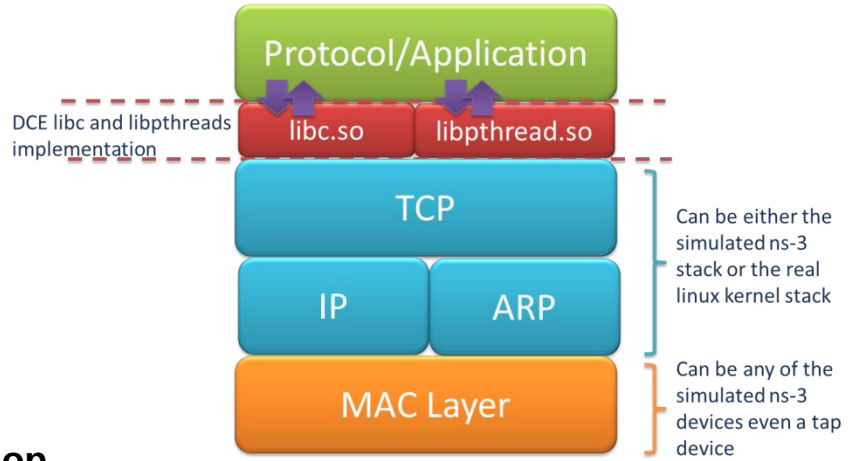
MNM Workshop
May 2015

Goals

- Lightweight virtualization of kernel and application processes, interconnected by simulated networks
- Benefits:
  - Implementation realism in controlled topologies or wireless environments
  - Model availability
  - Debugging a whole network within a single process
- Limitations:
  - Not as scalable as pure simulation
  - Tracing more limited
  - Configuration different

Direct Code Execution

- DCE/ns-3 framework requires the virtualization of a series of services
  - Multiple isolated instances of the same protocol on the same machine
- System calls are captured and treated by DCE
- Network stack protocols calls are captured and redirected
- To perform its work DCE re-implement the Linux program loader and parts of *libc* and *libpthread*



Direct Code Execution

- Developed by Mathieu Lacage and Frederic Urbani, INRIA, Hajime Tazaki (University of Tokyo)

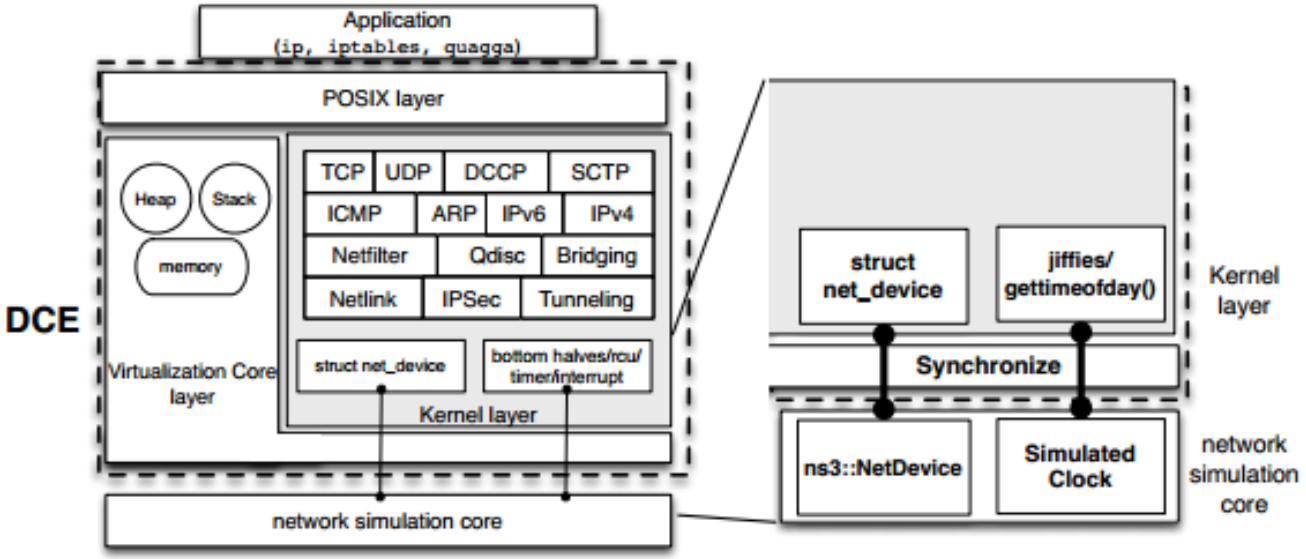


Figure 1: Architecture of Direct Code Execution. Kernel network devices and timers are synchronized with simulated NetDevice and clock.

Figure source: Direct Code Execution: Revisiting Library OS Architecture for Reproducible Network Experiments (CONEXT 13)

DCE modes

- DCE modes in context of possible approaches

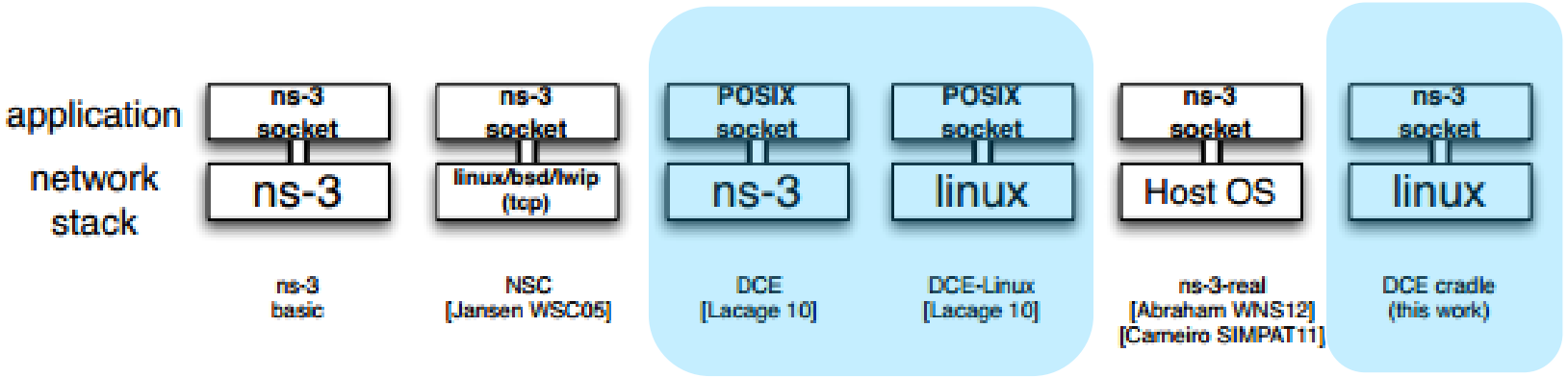


Figure 1: Current possible combinations of network stacks and applications.

Figure source: DCE Cradle: Simulate Network Protocols with Real Stacks for Better Realism, Tazaki et al, WNS3 2013.

DCE current version

DCE 1.5 released with ns-allinone-3.22

Recent additions

- Stream Control Transmission Protocol (SCTP) support for DCE
- FreeBSD version of network stack (experimental)
- support IPv6 sockets for DCE Cradle
- Support for Python bindings (partial)

- https://www.nsnam.org/docs/dce/manual/html/index.html

Paper references

- Direct Code Execution: Revisiting Library OS Architecture for Reproducible Network Experiments
  - Tazaki et al, CONEXT 2013
  - http://hal.archives-ouvertes.fr/docs/00/88/08/70/PDF/con013-hal.pdf
- DCE Cradle: Simulate Network Protocols with Real Stacks for Better Realism
  - Tazaki et al, WNS3 2013
  - http://hal.archives-ouvertes.fr/docs/00/78/15/91/PDF/wns3-2013.pdf

DCE example walkthrough

- hg clone http://code.nsnam.org/bake bake
- export BAKE_HOME=`pwd`/bake
- export PATH=$PATH:$BAKE_HOME
- export PYTHONPATH=$PYTHONPATH:$BAKE_HOME
- mkdir dce
- cd dce
- bake.py configure -e dce-ns3-1.5
- bake.py download
- bake.py build

Hands on, what do we need

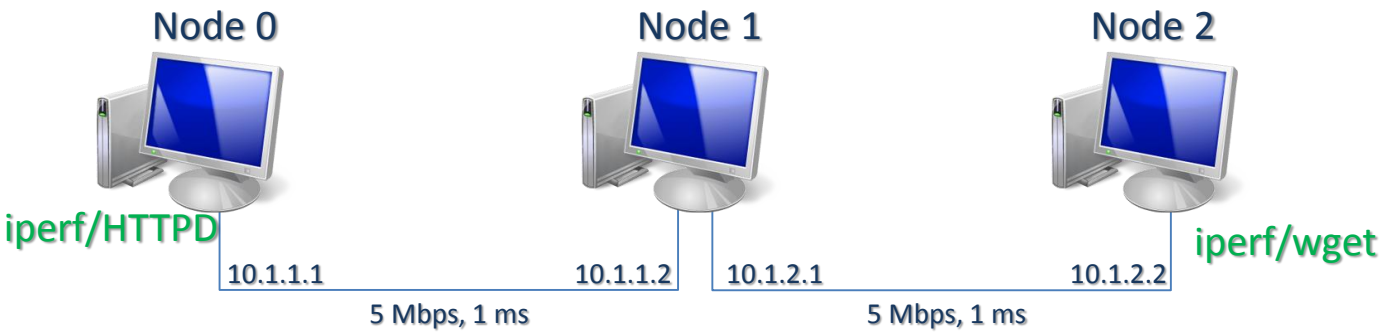
- **What do you need to start using the framework!**
  - ns-3 – The network simulator
    - http://www.nsnam.org/
  - DCE
    - http://www.nsnam.org/overview/projects/direct-code-execution/
  - Applications
    - iperf, wget, thttpd

  * All software must be re-compiled with –fpic and linked with –pie to generate the code with Position Independent Code (PIC) and permit context switch

- **To make things easier**
  - Bake – Installation tool
    - http://code.nsnam.org/bake
  - Mercurial – source control management tool
    - http://mercurial.selenic.com/
  - Python – for running bake
    - www.python.org

The shared scenario

- **The shared scenario is a simple three nodes network**



Step by step example
- What we need to do!

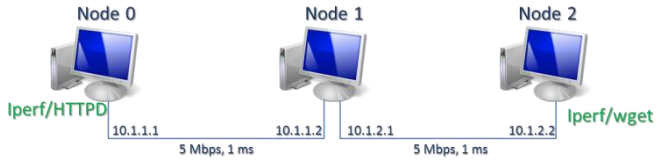
1. Create the nodes
2. Create stack
3. Create devices
4. Set addresses
5. Connect devices
6. Create DCE
7. Configuration the applications to run
8. Set start time for server and client
9. Set simulation time
10. Start simulation

Step by step example
- What we need to do!

1) Create the nodes
2) Create stack
3) Create devices
4) Set addresses
5) Connect devices
6) Create DCE
7) Configuration the applications to run
8) Set start time for server and client
9) Set simulation time
10) Start simulation

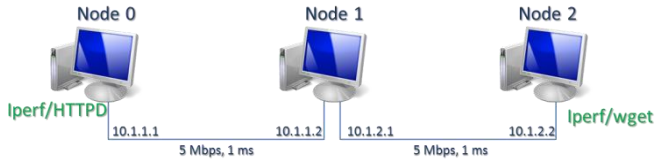
Standard ns-3 procedures
DCE specific

Step by step example
- iperf with ns-3 stack (I)




```
int main (int argc, char *argv[])
{
  // Node Container creation
  NodeContainer nodes;
  nodes.Create (3);

  // Linux stack creation
  InternetStackHelper stack;
  stack.Install (nodes);

  // For real time
  // GlobalValue::Bind ("SimulatorImplementationType", StringValue ("ns3::RealtimeSimulatorImpl"));
  // GlobalValue::Bind ("ChecksumEnabled", BooleanValue (true));

  // Device and channel  creation
  PointToPointHelper p2p;
  p2p.SetDeviceAttribute ("DataRate", StringValue ("5Mbps“));
  p2p.SetChannelAttribute ("Delay", StringValue ("1ms"));
```

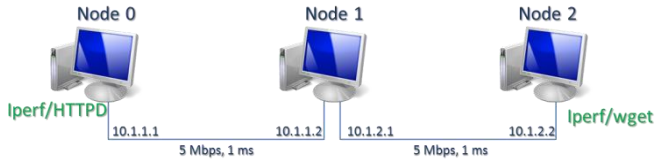
Step by step example
- iperf with ns-3 stack (II)




```
//  Node0-Node1 setup
 Ipv4AddressHelper address;
 address.SetBase ("10.1.1.0", "255.255.255.252"); //  Node0-Node1 addresses

  NetDeviceContainer devices;
 devices = p2p.Install (nodes.Get (0), nodes.Get (1)); // connecting nodes
 Ipv4InterfaceContainer interfaces = address.Assign (devices); //  assign addresses

 //  Node1-Node2 setup
 devices = p2p.Install (nodes.Get (1), nodes.Get (2)); // connecting nodes
 address.SetBase ("10.1.2.0", "255.255.255.252"); //  Node1-Node2 addresses
 interfaces = address.Assign (devices); //  assign addresses

 // setup ip routes
 Ipv4GlobalRoutingHelper::PopulateRoutingTables ();
```

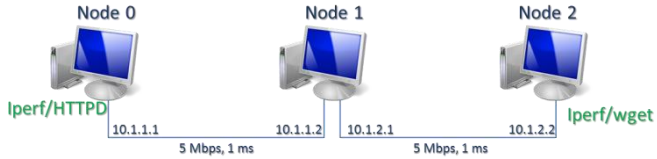
Step by step example - iperf with ns-3 stack (III)

```
DceManagerHelper dceManager;
dceManager.Install (nodes);

DceApplicationHelper dce;
ApplicationContainer apps;
 dce.SetStackSize (1 << 20); // 1MB stack

dce.SetBinary ("iperf"); // Launch iperf client on node 0
dce.ResetArguments ();  // clean arguments
dce.ResetEnvironment (); // clean environnent
dce.AddArgument ("-c");  // client
dce.AddArgument ("10.1.2.2"); //target machine address
dce.AddArgument ("-i"); // interval
dce.AddArgument ("1");
dce.AddArgument ("--time"); // how long
dce.AddArgument ("10");
 apps = dce.Install (nodes.Get (0)); //install application
apps.Start (Seconds (0.7)); //start at 0.7 simulation time
apps.Stop (Seconds (20)); //stop at 20s        simulation time

dce.SetBinary ("iperf"); // Launch iperf server on node 2
dce.ResetArguments (); // clean arguments
dce.ResetEnvironment (); // clean environnent
dce.AddArgument ("-s"); // server
dce.AddArgument ("-P"); // number of paralell servers
dce.AddArgument ("1");
apps = dce.Install (nodes.Get (2));
apps = dce.Install (nodes.Get (2));
apps.Start (Seconds (0.6));
```



DCE Setup

Step by step example
- iperf with ns-3 stack (IV)



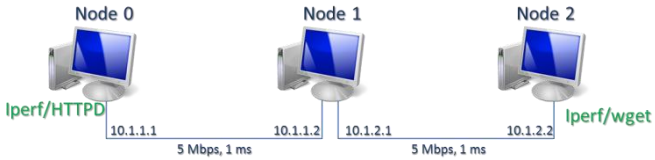

```
 // Simulation stop time
 Simulator::Stop (Seconds (40.0));

 // Run
 Simulator::Run ();

 // Stop
 Simulator::Destroy ();

 return 0;
}
```

Step by step example – iperf, ns-3

- **Generated**
  - elf-cache – program files
  - exitprocs – execution process information
  - files-0 files-2 – execution filesystem

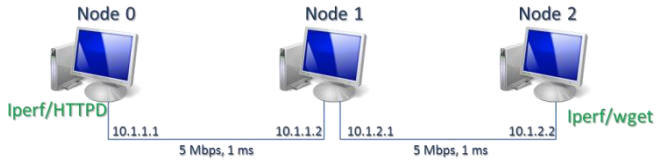
- **files-x**
  - var – “/root” of the machine
  - files-x/var/log/<pid>/
    - cmdline – command executed
    - status – execution information
    - stderr – standard error output
    - stdout – standard output
    - syslog – syslog output

Step by step example
- HTTP with ns-3 stack (I)




```
int main (int argc, char *argv[])
{
  // Node Container creation
  NodeContainer nodes;
  nodes.Create (3);

  // Linux stack creation
  InternetStackHelper stack;
  stack.Install (nodes);

  // For real time
  // GlobalValue::Bind ("SimulatorImplementationType", StringValue ("ns3::RealtimeSimulatorImpl"));
  // GlobalValue::Bind ("ChecksumEnabled", BooleanValue (true));

  // Device and channel  creation
  PointToPointHelper p2p;
  p2p.SetDeviceAttribute ("DataRate", StringValue ("5Mbps“));
  p2p.SetChannelAttribute ("Delay", StringValue ("1ms"));
```

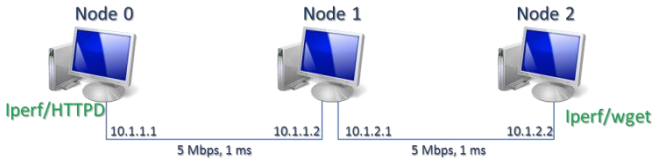
Step by step example
- HTTP with ns-3 stack (II)




```
//  Node0-Node1 setup
 Ipv4AddressHelper address;
 address.SetBase ("10.1.1.0", "255.255.255.252"); //  Node0-Node1 addresses

  NetDeviceContainer devices;
 devices = p2p.Install (nodes.Get (0), nodes.Get (1)); // connecting nodes
 Ipv4InterfaceContainer interfaces = address.Assign (devices); //  assign addresses

 //  Node1-Node2 setup
 devices = p2p.Install (nodes.Get (1), nodes.Get (2)); // connecting nodes
 address.SetBase ("10.1.2.0", "255.255.255.252"); //  Node1-Node2 addresses
 interfaces = address.Assign (devices); //  assign addresses

 // setup ip routes
 Ipv4GlobalRoutingHelper::PopulateRoutingTables ();
```

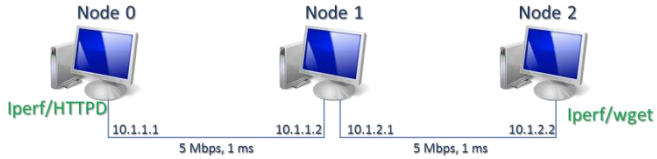
Step by step example
- HTTP with ns-3 stack (III)

```
// Launch the server HTTP
 dce.SetBinary ("thttpd");
 dce.ResetArguments ();  // clean arguments
 dce.ResetEnvironment (); // clean environnent
 dce.SetUid (1);  // Set httpd for super user execution
 dce.SetEuid (1);
 apps = dce.Install (nodes.Get (0)); // install http daemon
 apps.Start (Seconds (1)); // start time

 // Launch the client WGET
 dce.SetBinary ("wget");
dce.ResetArguments ();  // clean arguments
 dce.ResetEnvironment (); // clean environnent
 dce.AddArgument ("-r"); // recursive wget
 dce.AddArgument ("http://10.1.1.1/index.html");
 apps = dce.Install (nodes.Get (2));
 apps.Start (Seconds (2)); // start time
```



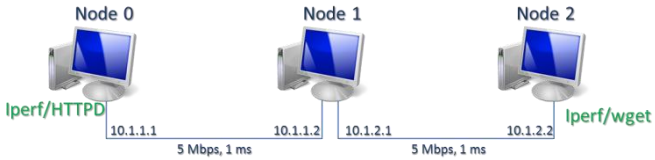
DCE Setup

Step by step example
- HTTP with ns-3 stack (IV)



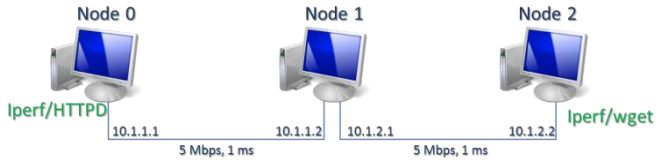

```
 // Simulation stop time
 Simulator::Stop (Seconds (40.0));

 // Run
 Simulator::Run ();

 // Stop
 Simulator::Destroy ();

 return 0;
}
```

Step by step example
- iperf with linux stack (I)




```
int main (int argc, char *argv[])
{
 // Node Container creation
 NodeContainer nodes;
 nodes.Create (3);

 // Linux stack creation
 dceManager.SetNetworkStack ("ns3::LinuxSocketFdFactory", "Library", StringValue ("liblinux.so"));
 LinuxStackHelper stack;
 stack.Install (nodes);

// For real time
 // GlobalValue::Bind ("SimulatorImplementationType", StringValue ("ns3::RealtimeSimulatorImpl"));
 // GlobalValue::Bind ("ChecksumEnabled", BooleanValue (true));

 // Device and channel  creation
 PointToPointHelper p2p;
 p2p.SetDeviceAttribute ("DataRate", StringValue ("5Mbps“));
 p2p.SetChannelAttribute ("Delay", StringValue ("1ms"));
```

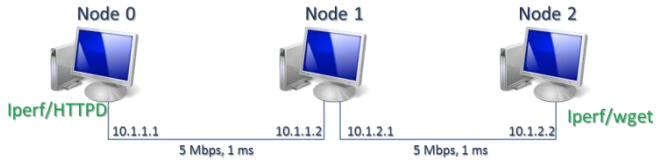
Step by step example
- iperf with linux stack (II)




```
//  Node0-Node1 setup
 Ipv4AddressHelper address;
 address.SetBase ("10.1.1.0", "255.255.255.252"); //  Node0-Node1 addresses

  NetDeviceContainer devices;
 devices = p2p.Install (nodes.Get (0), nodes.Get (1)); // connecting nodes
 Ipv4InterfaceContainer interfaces = address.Assign (devices); //  assign addresses

 //  Node1-Node2 setup
 devices = p2p.Install (nodes.Get (1), nodes.Get (2)); // connecting nodes
 address.SetBase ("10.1.2.0", "255.255.255.252"); //  Node1-Node2 addresses
 interfaces = address.Assign (devices); //  assign addresses

 // setup ip routes
 Ipv4GlobalRoutingHelper::PopulateRoutingTables ();
```

Step by step example
- iperf with linux stack (III)

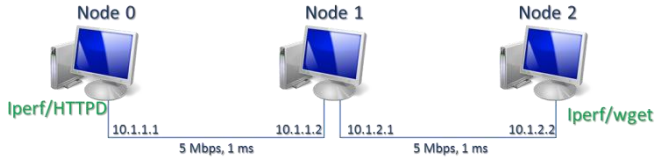
```
DceManagerHelper dceManager;
dceManager.Install (nodes);

DceApplicationHelper dce;
ApplicationContainer apps;
 dce.SetStackSize (1 << 20); // 1MB stack

dce.SetBinary ("iperf"); // Launch iperf client on node 0
dce.ResetArguments ();  // clean arguments
dce.ResetEnvironment (); // clean environnent
dce.AddArgument ("-c");  // client
dce.AddArgument ("10.1.2.2"); //target machine address
dce.AddArgument ("-i"); // interval
dce.AddArgument ("1");
dce.AddArgument ("--time"); // how long
dce.AddArgument ("10");
 apps = dce.Install (nodes.Get (0)); //install application
apps.Start (Seconds (0.7)); //start at 0.7 simulation time
apps.Stop (Seconds (20)); //stop at 20s        simulation time

dce.SetBinary ("iperf"); // Launch iperf server on node 2
dce.ResetArguments (); // clean arguments
dce.ResetEnvironment (); // clean environnent
dce.AddArgument ("-s"); // server
dce.AddArgument ("-P"); // number of paralell servers
dce.AddArgument ("1");
apps = dce.Install (nodes.Get (2));
apps = dce.Install (nodes.Get (2));
apps.Start (Seconds (0.6));
```



DCE Setup

(Similar to the ns-3 stack one)

Step by step example
- iperf with linux stack (IV)




```
 // Simulation stop time
 Simulator::Stop (Seconds (40.0));

 // Run
 Simulator::Run ();

 // Stop
 Simulator::Destroy ();

 return 0;
}
```