Experimentation with CCN

Daniel Camara, Frederic Urbani, Mathieu Lacage, Thierry Turletti, Walid Dabbous

INRIA, Planète

Sept, 13th 2012
Idealized Protocol Evaluation

Analytical analysis → Simulator → Testbed

In the wild → Large scale field → Small scale field
A Development Tool

- Debugging
- Validation
- Automated Testing (fast, deterministic)
A Development Tool

Development of Production Code:
Development of Production Code: Debugging
Development of Production Code:
  Debugging
  Validation
Development of Production Code:

- Debugging
- Validation
- Automated Testing (fast, deterministic)
Testbeds?

- Deployment too slow and unreliable
- Not completely deterministic
- Distributed debugging is hard
- Distributed tracing is hard

Camara, Urbani (INRIA)  Experimentation/CCN  Sept 2012
Testbeds?

No:

- Deployment too slow and unreliable
- Not completely deterministic
- Distributed debugging is hard
- Distributed tracing is hard
No:

Deployment too slow and unreliable
No:

- Deployment too slow and unreliable
- Not completely deterministic
No:

- Deployment too slow and unreliable
- Not completely deterministic
- Distributed debugging is hard
No:

- Deployment too slow and unreliable
- Not completely deterministic
- Distributed debugging is hard
- Distributed tracing is hard
Traditional Simulators?

No:
Traditional Simulators?

No: Not the real code
Proposed Solution

- Simulate the CCNx code before deployment
  - Reuse existing layer 1/2/3/4 models from ns-3
  - Trivial to deploy multiple network nodes
  - Easy whole-system debugging
  - Easy whole-system tracing and analysis
  - Optionally reuse kernel layer 3/4 stack
  - Much better scalability than testbeds/VMs
Proposed Solution

Simulate the CCNx code before deployment:

- Reuse existing layer 1/2/3/4 models from ns-3
- Trivial to deploy multiple network nodes
- Easy whole-system debugging
- Easy whole-system tracing and analysis
- Optionally reuse kernel layer 3/4 stack
- Much better scalability than testbeds/VMs
Proposed Solution

Simulate the CCNx code before deployment:
Reuse existing layer 1/2/3/4 models from ns-3
Simulate the CCNx code before deployment:
  Reuse existing layer 1/2/3/4 models from ns-3
  Trivial to deploy multiple network nodes
Proposed Solution

Simulate the CCNx code before deployment:
- Reuse existing layer 1/2/3/4 models from ns-3
- Trivial to deploy multiple network nodes
- Easy whole-system debugging
- Optionally reuse kernel layer 3/4 stack
- Much better scalability than testbeds/VMs
Proposed Solution

Simulate the CCNx code before deployment:
• Reuse existing layer 1/2/3/4 models from ns-3
• Trivial to deploy multiple network nodes
• Easy whole-system debugging
• Easy whole-system tracing and analysis
Simulate the CCNx code before deployment:

- Reuse existing layer 1/2/3/4 models from ns-3
- Trivial to deploy multiple network nodes
- Easy whole-system debugging
- Easy whole-system tracing and analysis
- Optionally reuse kernel layer 3/4 stack

Camara, Urbani (INRIA)  Experimentation/CCN  Sept 2012
Proposed Solution

Simulate the CCNx code before deployment:
- Reuse existing layer 1/2/3/4 models from ns-3
- Trivial to deploy multiple network nodes
- Easy whole-system debugging
- Easy whole-system tracing and analysis
- Optionally reuse kernel layer 3/4 stack
- Much better scalability than testbeds/VMs
How do I use it?

Roughly:

▶ Download/install ns-3
▶ Download/install ns-3 DCE module
▶ Recompile CCNx with magic option
▶ Write simulation script
▶ Run
▶ Analyse traces

The details: http://goo.gl/yfgwZ
How do I use it?

Roughly:

- Download/install ns-3
- Download/install ns-3 DCE module
- Recompile CCNx with magic option
- Write simulation script
- Run
- Analyse traces

The details: [http://goo.gl/yfgwZ](http://goo.gl/yfgwZ)
How do I use it?

Roughly:

- Download/install ns-3
- Download/install ns-3 DCE module
- Recompile CCNx with magic option
- Write simulation script
- Run
- Analyse traces

The details: http://goo.gl/yfgwZ
What we would like to learn

If you tried DCE/CCNx:
▶ What sucks about it?
▶ How can we improve it?
▶ How can we write more test scripts for CCNx?

If you did not try it:
▶ Why?
▶ What is missing to make you try it?
What we would like to learn

If you tried DCE/CCNx:
  What sucks about it?
  How can we improve it?
  How can we write more test scripts for CCNx?

If you did not try it:
  Why?
  What is missing to make you try it?
What we would like to learn

If you tried DCE/CCNx:
  ▶ What sucks about it?
  ▶ How can we improve it?
  ▶ How can we write more test scripts for CCNx?

If you did not try it:
  ▶ Why?
  ▶ What is missing to make you try it?
Thank you!

Contact:

daniel.camara@inria.fr
frederic.urbani@inria.fr
mathieu.lacage@cutebugs.net
### Related Work

<table>
<thead>
<tr>
<th></th>
<th>NDN over ONL</th>
<th>NEPI</th>
<th>ccnSim</th>
<th>CCNPL-Sim</th>
<th>DCE</th>
<th>ndnSim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>testbed</td>
<td>testbed</td>
<td>sim</td>
<td>sim</td>
<td>sim</td>
<td>sim</td>
</tr>
<tr>
<td>Real code</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Scalability</td>
<td>−</td>
<td>−</td>
<td>+++</td>
<td>?</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Deployment</td>
<td>−</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Debugger</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Tracing</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
Scalability

Memory:
- Fixed, 1MB/node ccnx
- Fixed, 0.02MB/node ns-3 TCP/IP
- Fixed, 0.5MB/node linux TCP/IP
- Linear in number of bytes of packets in flight

CPU:
- Real time limit: 20 nodes, 200K file transfer