ns-3 Direct Code Execution
Objective Scenario
Objective Scenario

VLC Server

Wifi STA

Router

VLC Client

Router

Wifi AP

ns-3 DCE
Study the impact of wifi routing on video stream quality
Goals

Study the impact of wifi routing on video stream quality

Develop new wifi adhoc routing protocols
Need simulations:
- Reproducibility
- Debuggability
- Testability

Need real-world experiments:
- Wireless medium realism

BUT:
- Must maintain two implementations
Problem

Need simulations:
  Reproducibility
  Debuggability
  Testability

Need real-world experiments:
  Wireless medium realism
Need simulations:
  Reproducibility
  Debuggability
  Testability
Need real-world experiments:
  Wireless medium realism
BUT:
  Must maintain two implementations
Manual Modifications

Convert global variables in arrays
Manual Modifications

Convert global variables in arrays
Convert system calls in simulation calls

BUT, does not scale:
▶ Painful to do once
▶ Impossible to do for software updates

The solution: Direct Code Execution
▶ Automate global variable virtualization
▶ Automate system call redirection
▶ Provide simulation system call replacements
Convert global variables in arrays
Convert system calls in simulation calls
BUT, does not scale:
  Painful to do once
  Impossible to do for software updates
Convert global variables in arrays
Convert system calls in simulation calls
BUT, does not scale:
  Painful to do once
  Impossible to do for software updates
The solution: Direct Code Execution
  Automate global variable virtualization
  Automate system call redirection
  Provide simulation system call replacements
Related Work

Network Simulation Cradle:
- Automated source modifications for C code
- Hard to extend to C++
Related Work

Network Simulation Cradle:
- Automated source modifications for C code
- Hard to extend to C++

Weaves:
- Automated textual assembly modifications
- Does not work in practice
Related Work

Network Simulation Cradle:
Automated source modifications for C code
Hard to extend to C++

Weaves:
Automated textual assembly modifications
Does not work in practice

COOJA:
Automated memory virtualization
Slow
Adhoc ELF Loader:
  Fast
  Automated memory virtualization
  Automated system call redirection
Adhoc ELF Loader:
  Fast
  Automated memory virtualization
  Automated system call redirection
Userspace system calls
Adhoc ELF Loader:
  Fast
  Automated memory virtualization
  Automated system call redirection
Userspace system calls
Kernelspace system calls
Scenario:
Loader Performance

Average packets per wall clock second vs. Number of nodes for different simulators:
- Cooja
- dlmopen
- Lacage (INRIA)
- ns-3 DCE
Loader Performance

![Graph showing Loader Performance](image)

- **Cooja**
- **dlmopen**

Average memory usage (KB) vs. Number of nodes

Lacage (INRIA)
ns-3 DCE
GEC9
System Performance

Average packets per wall clock second

Number of nodes
dce-none
dce-user
dce-user+kernel

Lacage (INRIA)
ns-3 DCE
System Performance

![Graph showing system performance](Image)

- **Average memory usage (KB)**
- **Number of nodes**
- **dce-none**
- **dce-user**
- **dce-user+kernel**

Lacage (INRIA)

ns-3 DCE

GEC9  12 / 14
Reuse existing protocol implementations:

Userspace: ping, traceroute, quagga, etc.
Kernelspace: IP, TCP, etc.
Reuse existing protocol implementations:

Userspace: ping, traceroute, quagga, etc.
Kernelspace: IP, TCP, etc.

Debugging platform: Single debugger controls all protocol instances
Reuse existing protocol implementations:
  Userspace: ping, traceroute, quagga, etc.
  Kernelspace: IP, TCP, etc.

Debugging platform: Single debugger controls all protocol instances

Development platform
Conclusion

Reuse existing protocol implementations:
   Userspace: ping, traceroute, quagga, etc.
   Kernelspace: IP, TCP, etc.

Debugging platform: Single debugger controls all protocol instances

Development platform

Test platform
Future Work

Improve userspace API coverage:
   fork, wait, exec

Add X11 connection forwarding

More testing

Documentation

Write paper