Global Environment for Network Innovation (GENI)

Trema

An open source OpenFlow controller platform

HIDEyuki Shimonishi, Yasunobu Chiba, Yasuhito Takamiya, Kazushi Sugyo
System Platforms Research Laboratories, NEC Corporation

Try Now !

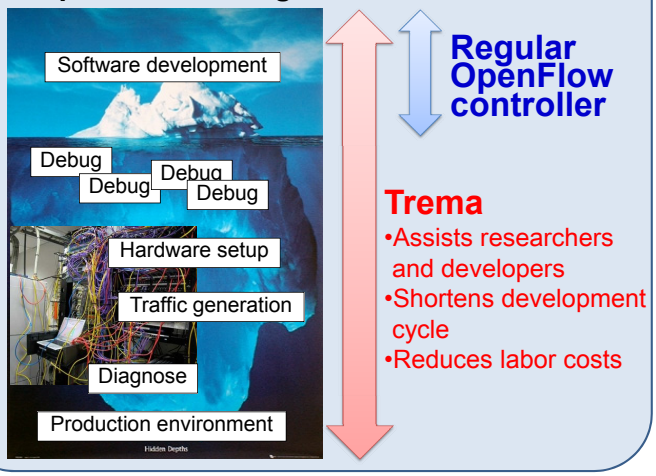
```
$ git clone git://github.com/trema/trema.git
$ ./trema/demo.rb
```

* Tested with Debian GNU/Linux and Ubuntu Linux.

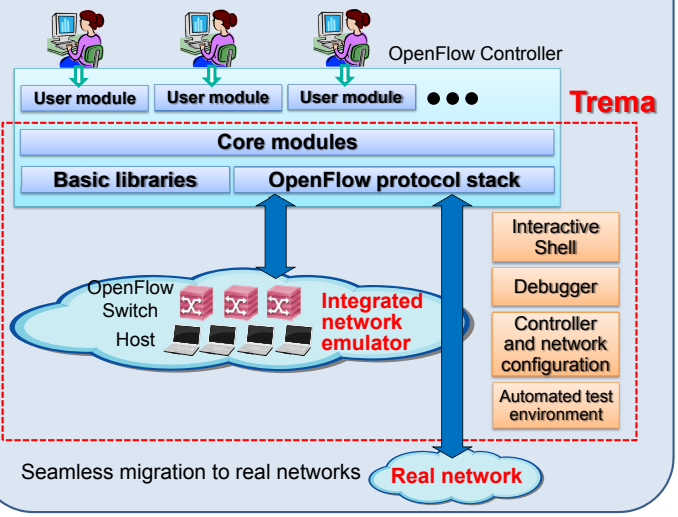
What is Trema?

- A free OpenFlow controller platform (GPL v2)
  - Assists anyone who wants to develop his/her own Openflow controller
  - Not targeted for any specific Openflow controller implementation
- Allows to implement OpenFlow controllers in:
  - **C** (fully supported)
  - **Ruby** (partially supported. planned to be fully supported by Q3/2011)
- Contact
  - Mailing list: **trema-dev@googlegroups.com** / twitter: **@trema_news**

Scope



Components



Repeater HUB emulation in Ruby

```
class RepeaterHub < Trema::Controller
  def packet_in message
    send_flow_mod_add(
      message.datapath_id,
      :match => Match.from( message ),
      :actions => Trema::ActionOutput.new( OFPP_FLOOD )
    )
    send_packet_out(
      message.datapath_id,
      :buffer_id => message.buffer_id,
      :actions => Trema::ActionOutput.new( OFPP_FLOOD ),
      :data => message.buffered? ? nil : message.data
    )
  end
end
```

Repeater HUB emulation in C

```
static void handle_packet_in( packet_in message ) {
  openflow_actions *actions = create_actions();
  append_action_output( actions, OFPP_FLOOD, UINT16_MAX );
  struct ofp_match match;
  set_match_from_packet( &match, message.in_port, 0, message.data );
  buffer *flow_mod = create_flow_mod( ... );
  send_openflow_message( message.datapath_id, flow_mod );
  free_buffer( flow_mod );
  buffer *packet_out = create_packet_out( ... );
  send_openflow_message( message.datapath_id, packet_out );
  free_buffer( packet_out );
  delete_actions( actions );
}

int main( int argc, char *argv[] ) {
  init_trema( &argc, &argv );
  set_packet_in_handler( handle_packet_in, NULL );
  start_trema();
  return 0;
}
```