Case Study in Using Virtualization as an Architectural Tool

Dagstuhl Network Virtualization Workshop, September 2008

Jari Arkko
jari.arkko@ericsson.com
Ericsson Research

Background

- Virtualization has been used to create private networks, use slices of experiment platforms, ...
- But could virtualization be a tool in a new Internet architecture by its own right?
- Observe where innovation is in the current Internet
- The hypothesis is that this situation could be different in a future architecture if it supported virtualization

My Approach

Attempt to test the hypothesis through examples

- Is virtualization useful in these examples?
- What does it have to do to be useful?
- Has virtualization already been used?
- Understand better the limits of virtualization

Examples:

- Improvements to routing scalability
- Introduction of IPv6

Example 1: Routing Scalability

- Recent concern in the growth of table size and dynamics in the global BGP routing tables

- This problem can be solved in several ways:
  - Better microelectronics in routers
  - Better algorithms managing the electronics
  - Evolutionary changes in BGP
  - Revolutionary changes in routing protocols
  - Revolutionary changes in Internet architecture

Routing Scalability, Cont'd

How do these work with virtualization:

- Better microelectronics in routers
  - Cannot do this
- Better algorithms managing the electronics
  - This might be possible
  - But it is likely that the algorithms related to RIB-to-FIB updates are very hardware specific, for instance

Routing Scalability, Cont'd

- Evolutionary changes in BGP
  - Given that these are evolutionary, normal protocol engineering tools in BGP should suffice?
    - Not always easy, have to work through older devices
  - If it was possible to deploy new code in the entire Internet via virtualization, would it help?
  - But note that this would be *changes to an existing network*, not new code in a different slice
  - Unlikely in the global scale, but possible per-domain
  - Separating the commodity substrate from the intelligence is still useful for faster evolution

Routing Scalability, Cont'd

- Revolutionary changes
  - A new routing protocol, separation of addresses for core and the edge, or a full idloc separation
  - Each one of these is easily deployable by itself
    - New protocols run in new port numbers,
    - A mesh of tunnels crosses the old Internet to connect the new parts to each other
    - New hosts can behave in new ways
  - The tricky part is making it possible to communicate between the new and the old Internets

    ("I hear there's rumors on the, uh, Internets" -- G. W. Bush)
  - Inserting new code/entries everywhere is contrary to the idea of relieving pain in the forwarding table

Example 2: IPv6 Introduction

- Protocol-wise ready in late 1990s
- Today widely but not universally implemented in hosts and routers; not very widely turned on
- Would virtualization have helped?
  - Early adopters using IPv6 in their own networks
  - Full-blown adoption in ISP networks
  - Ability to communicate between the two Internets

IPv6 Introduction, Cont'd

- Early adopters using IPv6 in their own networks
  - In fact, it did help (virtual tunnels)
- Full-blown adoption in ISP networks
  - Adding IPv6 to router products – Yes
  - Reaching feature & speed parity with IPv4 – ?
  - ISP peering policy and incentives – ?
- Ability to communicate between the two Internets
  - It would be easier to deploy translators between network edges – but this requires advanced functionality

Conclusions

- Existing virtualization tools have been used
- Additional tools useful, but not silver bullets
  - Modularization to a dumb substrate and an intelligent server speeds up innovation
- The crux is what abstraction you provide
  - Mere separation is insufficient – need interworking
  - Cannot simply ask to do any random act – very dependent on hardware capabilities
    - Random packet attack
  - How does all this work with peering, TE, etc
  - Supporting ”The Internet” vs. a private network
  - Most interesting cases involve non-trivial operations