Interoperable Convergence of Storage, Networking and Computation

🔥🔥 The Fire Down Below! 🔥🔥

Micah Beck

Terry Moore, Piotr Luszczek & Anthony Denalis
Innovative Computing Lab.

University of Tennessee, Knoxville

43rd ORAP Forum 3/19/2019
Motivation and Background

- 20 yrs: Web caches to prestage content (TN Cachebox)
- 15 yrs: Logistical Networking as async. communication
- Design methodology: follow patterns of Internet and Unix
- Architectural analysis: very confused, much controversy
- Experiment, deploy, describe... try, try, try to understand!
- Slow realization: storage, processing *implement* layer 3
- 2016: some formal insight into the hourglass model
Interoperation and Innovation

“What is lacking … is a common, open, underlying ‘platform’, analogous to … the Internet or Web, allowing applications and services to be developed as modular, extensible, interoperable components. To achieve the level of interoperation and innovation… that we have seen in the Internet will require investment in the basic research and development of an analogous open platform for intelligent infrastructure.…”

[City-Scale Intelligent Systems and Platforms, Nahrstedt 2017]
Historical Context: Divergence of IT silos

- Starting with gates, wires and cores
- Modeled by Boolean operators and variables
- Divergence of communication, transformation, persistence
- Giving rise to networking, computation and storage silos
- But optimization & service creation work across silos
  - Active Networking, Web Caching, Content Delivery
  - Grid, SDN, Network Function Virtualization, Workflows
- This talk proposes an architectural strategy for convergence
Legacy Silos and Overlay Convergence

Overlay Convergence

Network

Storage

Compute
The Hourglass vs. The Anvil
Publications


• ”On the Hourglass Model, End-to-End Arguments and Deployment Scalability”, Micah Beck, To Appear in Communications of the ACM in 2019 (http://philsci-archive.pitt.edu/id/eprint/12626)
Our Proposal: A common abstraction of the IT node

- Convergence can occur at many levels
  - Operating system kernel interface (e.g., file systems, sockets)
  - Application overlay environments (e.g., PVM, Grid, Workflows)

- Convergence at a high level preserves silo abstractions
  - Reliability, performance, fairness, permanence, identity, accountability

- Convergence at a high level limits interop. and ubiquity
  - Resources at the bottom of the stack are accessed through the top

- We propose a converged common abstraction of buffer management, transformation & transfer at the local layer.
Buffer Interoperability: Linux `sendfile()` system call
Interoperability: The spanning layer

- The spanning layer as the basis of interoperability
  “Certain protocols are designed with the specific purpose of bridging differences at the lower layers… Such a layer is called a ‘spanning layer’…” David. Clark, Interoperation, Open Interfaces, and Protocol Architecture, 1997
- \( O(N^2) \) translations, perhaps imperfect
- A common spanning layer enables “network effects”
- The “thin waist” of the hourglass
- “Interoperable convergence”: full generality, performance
Deployment Scalability: Simple, generic, limited

- Overcoming barriers: technological, geographical, admin.
- Successful infrastructure interfaces scale “by design”
  - Not regulation or market dominance
- The strategy of the Internet: converge at a lower layer
  - Expose underlying resources
  - Enable flexibility and heterogeneity at higher layers
- Unix also followed a “lowering” design strategy
  - Saltzer, Clark, Reed, Thompson & Ritchie all worked on MULTICS
  - Internet and Unix are the two examples of deployment scalability
Logical Weakness: The Hourglass Theorem (Beck 2019)

• Competing notions of “thinness” at the spanning layer
• “Logical weakness” may exert a controlling effect
  ○ A weak spanning layer implies more implementations
  ○ But a weak spanning layer implies fewer applications!
• Recipe for deployment scalability:
  1. Select “necessary application requirements”
  2. Design the weakest spanning layer that satisfies them!
     Best effort, bounded lifetime, cycles, memory, block size, MTU, …
Heterogeneity: Lowering the spanning layer

• Ossification: The dark side of spanning layer ubiquity
  ○ Necessary uniformity at the spanning layer
  ○ Resources poorly exposed are not accessible
• IT nodes are computers: processors, storage, transfer
• In digital systems high level flow/process/file abstractions are implemented as sequences of buffer-to-buffer steps
• Minimal abstraction: operations on common buffer model
The Hourglass vs. The Anvil
Exposed Buffer Processing

• Fundamental Buffer Management
  ○ Local or in neighbor
  ○ Weak: best effort with limits imposed by node owner

• Allocate/delete buffer lease

• Write/Read to/from/xfer-between buffer(s)

• Apply operation to transform buffer(s)
  ○ Semi-static set of (weak, limited) operations available
  ○ An operation can be a VM (taking code as input)
“… memory locations … are just wires turned sideways in time”

Dan Hillis, 1982,
Why Computer Science is No Good
Exposing Buffers Implies Exposing “Topology”

• Stateless networking hides the existence of buffers
• Exposing buffers requires their location to be exposed.
• “Topology” of converged network: static & dynamic availability of bandwidth, storage and processing
• Effective management of workflows requires knowledge (and perhaps control) over generalized topology
• Internet architecture: apps are not “topologically informed”
Example: Content Delivery

- **Unicast datagram delivery and point-to-multipoint services**
  - Redundant point-to-point transmissions
  - Bottlenecks overload local network and server resources
  - Poor user performance or denial of service ("Slashdot effect")

- **Content Delivery employs caching & server replication**
  - A proprietary converged distributed platform “at application layer”
  - Specialized, expensive, difficult to deploy and operate
  - Layering violation – an overlay built out of pieces of the Internet stack
Application: Streaming in Workflows

• Classic workflow model allocates resources at high level
  ○ Storage: files
  ○ Communication: file transfer (localization)
  ○ Computation: jobs
• Actual requirements can be addressed more flexibly
  ○ “Run readiness” – no blocking on I/O
  ○ Localization is a hack – what if local file server or network is congested?
  ○ Streaming requires new high level abstractions – extend POSIX?
• Go the other way: Build streams out of buffer ops
Persistent processes vs. EBP ops

manager

data movement ops
Application: Edge processing

- Data deluge at the edge
  - Sensors and many other distributed data producers
  - Application-specific data volume reduction is required
- Thin clients require shared infrastructure
- Administration & policy are major roadblocks
- Trust, fair contention, cooperative sharing are enablers
- Remember: It works with LAN and OS resources
Example: Record filtering

1. FILTER <inbuf> <rwid> <rcnt> <fpos> <fieldwid> <val>

2. consider input buffer <inbuf> as an array of <rcnt> records of width <rwid> bits with a field starting at bit position <fpos> of width <fwid>

3. eliminate all records whose value in that field is not equal to <val>

4. Records generated by sensors at edge nodes, filling buffers

5. Continuously issued FILTER operations implement filtering “process”
Persistent processes vs. EBP ops
“Reality is Best Effort”

• At the low level, every workflow is built up out of simple, limited, faulty operations on buffers
• The total state of the workflow is a “macro process”
• Jobs, file storage and file transfers are built by a stack
  ○ A datagram is a buffer migrating across nodes
  ○ A storage system works hard to compute the identity function
  ○ A job executes one time slice at a time
• It is all ”dependence flow” (ordered imperative ops)
What about high level “requirements”?

- Security: moats and bridges
- Low latency transmission
- Highly synchronized computation
- Permanent, corruption-free storage
- Identity and accountability, metering and billing
- Overlay implementation: The Data Logistics Toolkit, Lead PI Martin Swany, Indiana University (data-logistics.org).
Interoperability: Edge through Net Core to Center?
Conclusions

- Divergence is a historical artifact, not inevitable
- Optimization across silos is necessary but challenging
- Convergence at local layer enables generality, innovation
- Heterogeneity is key to flexible global services
- Locality of access enables control and flexibility
Thank you!