MOSAIC: Unified Declarative Platform for Dynamic Overlay Composition

Yun Mao\textsuperscript{1,2}, Boon Thau Loo\textsuperscript{1}, Zachary Ives\textsuperscript{1}, and Jonathan M. Smith\textsuperscript{1}

University of Pennsylvania, AT&T Research
Challenges to Today’s Internet

• New applications demand new capabilities
  – Mobility, quality of service, content-based routing, anycast, multicast, …
  – Many applications require more than one capability

• Challenges
  – Unwanted and harmful traffic
  – Complexity and fragility of inter-domain routing

• Hard to address in the current architecture
  – Changing the core is hard
  – Protocols are deeply coupled to their implementations
Overlay Networks

- Application-level networks that achieve new functionality without changing the infrastructure:

  - **Resiliency**: RON, SOSR, DHARMA
  - **Mobility**: RMA, HIP, …
  - **Scalable Lookup**: Chord, Pastry, Tapestry, …
  - **Content Distribution**: Akamai, CoralCDN, …
  - **Security**: SOS, OverDoSe, …
  - **Multicast**: Overcast, ESM

- Do not fully address challenges and evolutionary needs
- Limited to application-specific domains
- Not easy to “mix-and-match” to support new applications
Example 1: Alice&Bob

Challenge 1: mobility

Alice at School

Bob

Alice at Home
Example 1: Alice&Bob

Challenge 2: lack of public IP address

Bob

NAT box

Alice at Home
Example 1: Alice&Bob

- Each individual challenge has solutions based on overlays, but none of them solve all the problems at once.
- Change in environment (trust level, connectivity, etc) may invalidate chosen overlay.
Example 2: Distributed Hash Table (DHT) with Network Failures

- Intermittent network failures result in broken return paths, and other issues
- Layer DHT over a resilient overlay can help!
Goals of Overlay Composition

• Speed-up network evolution:
  – Novel application-specific networks built on multiple overlay compositions (bridging / layering)
  – Component reusability
  – Dynamic adaptation: modify components as requirements or environment changes.

• Support custom application needs. E.g.,
  – Alice&Bob example:
    • i3+RON+bridges = mobility + reliability + NAT traversal
  – DHT in failure-prone networks
    • Chord DHT over RON = better lookup performance
  – Secure mobility:
    • i3 over secure overlay = mobility + eavesdropping prevention
  – ...

MOSAIC Approach

- Declarative framework for **rapidly prototyping** and **composing** overlay networks, and **dynamically changing** the compositions at runtime
- Leverages declarative networking [SIGCOMM ’05, SOSP ’05]:
  - Declarative specifications of networks using a distributed database query language
  - Distributed query engine executes specifications to implement network protocols
- Key advantages:
  - Compact and high-level representation of protocols
  - Orders of magnitude reduction in code size
  - Easy customization, sharing, and composability
MOSAIC Overview

Composition
Service Broker

Directory
Service

Composition Specification

Declarative Overlay

Declarative Overlay
Composition Specification

The composition graph of the Alice&Bob example:

- **i3**: Internet Indirection Infrastructure (for NAT and mobility)
- **RON**: Resilient Overlay Network

---

AliceNet → RON → BobNet

Bridge via Alice’s gateway

Bridge via Bob’s gateway

Layer over IP

Layer over the bridged network
Composition Specifications to Implementation

• Validation stage:
  – **Bridging**: two networks share common physical nodes
  – **Layering**: nodes of underlay are a super set of the overlay

• Compilation stage:
  – Declarative implementation of overlays
  – Generate the “glue code” as declarative networking rules
  – Ship rules to physical nodes for execution on a declarative networking engine (P2, http://p2.cs.berkeley.edu)
Background: Declarative Networking

**Network Datalog:** a distributed query language for networks

- **R1:** `reachable(\(@S,D\)) :- link(\(@S,D\))`  
  - For all nodes \(S, D\), there is a link from node \(S\) to node \(D\).  

- **R2:** `reachable(\(@S,D\)) :- link(\(@S,Z\)), reachable(\(@Z,D\))`  
  - If there is a link from \(S\) to \(Z\), and \(Z\) can reach \(D\), then \(S\) can reach \(D\).

Symbol definitions:

- `link(a,b)` – “there is a link from node \(a\) to node \(b\)”
- `reachable(a,b)` – “node \(a\) can reach node \(b\)”
Background: Declarative Networking

**Network Datalog:** a distributed query language for networks

R1: reachable(@S,D) :- link(@S,D)

R2: reachable(@S,D) :- link(@S,Z), reachable(@Z,D)

“For all nodes S,D and Z,
If there is a link from S to Z, AND Z can reach D, then S can reach D”.

<table>
<thead>
<tr>
<th>Input neighbor table:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-b-c</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output reachable table:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-b-c</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>link @S D</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-b-c</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>link @S D</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-b-c</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>link @S D</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-b-c</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>reachable (r1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-b-c</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>reachable (r2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-b-c</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>reachable (r2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-b-c</td>
</tr>
</tbody>
</table>
Large Library of Declarative Protocols

• Example implementations to date:
  – Routing protocols (DV, LS, DSR, AODV, OLSR, etc.)
  – Chord Distributed Hash Table
  – Resilient overlay network (RON)
  – Internet Indirection Infrastructure (i3)
  – Others: sensor networking protocols, replication, snapshot, fault tolerance protocols

• Language extensions:
  – Logical location specifiers (not just IP addresses)
  – Composable views (grouping rules together as components)
  – Legacy application support (via tunneling)
Composition Example: Alice & Bob

- Composition – 69 rules
  - Chord DHT – 35 rules, i3 – 16 rules, RON – 11 rules
  - Auto-generated composition “glue” – 7 rules (for layering and bridging)
Alice’s Initial State

<table>
<thead>
<tr>
<th>OverlayID</th>
<th>Overlay Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice_net</td>
<td>10.0.0.2</td>
</tr>
</tbody>
</table>

Table `netAddress`

<table>
<thead>
<tr>
<th>OverlayID</th>
<th>Overlay Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice_net</td>
<td>10.0.0.1</td>
</tr>
<tr>
<td>0</td>
<td>12.34.56.78</td>
</tr>
</tbody>
</table>

Table `netAddress`
Create RON

<table>
<thead>
<tr>
<th>OverlayID</th>
<th>Overlay Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice_net</td>
<td>10.0.0.2</td>
</tr>
</tbody>
</table>

Alice's Gateway

<table>
<thead>
<tr>
<th>OverlayID</th>
<th>Overlay Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice_net</td>
<td>10.0.0.1</td>
</tr>
<tr>
<td>0</td>
<td>12.34.56.78</td>
</tr>
<tr>
<td>ron</td>
<td>ron::12.34.56.78</td>
</tr>
</tbody>
</table>

RON over IP:
c1 underlay(ron,A):-
  netAddress(0,A).
Create bridge1

<table>
<thead>
<tr>
<th>OverlayID</th>
<th>Overlay Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice_net</td>
<td>10.0.0.2</td>
</tr>
<tr>
<td>bridge1</td>
<td>sr::[ron::12.34.56.78, 10.0.0.2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OverlayID</th>
<th>Overlay Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice_net</td>
<td>10.0.0.1</td>
</tr>
<tr>
<td>0</td>
<td>12.34.56.78</td>
</tr>
<tr>
<td>ron</td>
<td>ron::12.34.56.78</td>
</tr>
</tbody>
</table>

RON over IP:
\[
\text{cl underlay}(\text{ron}, A) :- \\
\text{netAddress}(0, A).
\]
Create i3

i3 over the bridged network:
\[
\text{c2} \ \text{underlay}(\text{i3 oid, A}) : - \\
\text{netAddress}(\text{bridge1, A}).
\]

RON over IP:
\[
\text{c1} \ \text{underlay}(\text{ron, A}) : - \\
\text{netAddress}(0, \text{A}).
\]

<table>
<thead>
<tr>
<th>OverlayID</th>
<th>Overlay Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice_net</td>
<td>10.0.0.2</td>
</tr>
<tr>
<td>bridge1</td>
<td>sr::[ron::12.34.56.78, 10.0.0.2]</td>
</tr>
<tr>
<td>i3</td>
<td>i3::alice_id</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OverlayID</th>
<th>Overlay Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice_net</td>
<td>10.0.0.1</td>
</tr>
<tr>
<td>0</td>
<td>12.34.56.78</td>
</tr>
<tr>
<td>ron</td>
<td>ron::12.34.56.78</td>
</tr>
</tbody>
</table>
Composed Network

Sender

i3::alice_id

sr::[ron::12.34.56.78, 10.0.0.2]

Intermediate RON node

Receiver

10.0.0.2

BobNet
Bob

BobNet
Bob’s gateway

IP

IP

AliceNet
Alice’s gateway
12.34.56.78

AliceNet
Alice

IP

IP

Alice’s gateway
12.34.56.78

Alice

10.0.0.2
Implementation

• Compilation to distributed dataflows
  – Similar to Click modular router
  – Additional support for relational operators, encapsulation/de-encapsulation, and legacy application support via tun device

• Evaluation:
  – Performance benchmarks in a LAN
  – Wide-area composition evaluations on PlanetLab
Experiment 1: Mobility + NAT + Reliability

PlanetLab nodes for i3 and RON

Alice in New Jersey
Bob in Utah

Composition achieves new functionalities at low performance overhead.
Dynamic Composition

• Underlays are specified *logically* rather than hard-coded
• Bind (or rebind) underlying network address
  – runtime binding = dynamic composition

\[
\text{s0 underlay(chord_oid}, \text{A}) :- \text{netAddress(0,}\text{A}).
\]
\[
\text{s1 underlay(chord_oid}, \text{A}) :- \text{netAddress(OID,}\text{A}),
\text{switchUnderlay(chord_oid}, \text{OID}).
\]
\[
\text{s2 underlay(ron_oid}, \text{A}) :- \text{netAddress(0,}\text{A}).
\]
Experiment 2: Dynamic Composition

Layering over RON improves DHT lookup accuracy, and can be composed dynamically.
Conclusions

• Contributions:
  – MOSAIC: A unified declarative platform for dynamic network composition
  – Leverages declarative networking techniques
  – Dynamic composition capabilities
  – Proof-of-concept deployment on PlanetLab

• Ongoing and future work:
  – Hybrid adaptable MANET Routing (SIGCOMM PRESTO ’08)
  – Declarative network verification (PADL ‘09)
    • Protocol reasoning, and in future, e2e composition properties
A mosaic is a larger pattern or picture constructed with small pieces of colored glass, stone, or other material.