UNIVERSITY OF CAMBRIDGE

A Data-Oriented (and Beyond) Network Architecture

Authors: T. Koponen, M. Chawla, B. Chun. K. Kim, S. Shenker, A. Ermolinskiy, I. Stoica

Presenter: Yi Liu

Feb 3, 2011

Introduction

- Applications shift from host-centric to data-centric
- User cares about what the content is instead of where to find it.
- Data-Oriented Network Architecture (DONA): a clean state redesign of Internet naming and name resolution.

User relevant issues

- **Persistence:**

  A data or service name remains valid as long as the data or service is available.

- **Availability:**

  Access to data and service should be reliable and have low-latency.

- **Authenticity:**

  Data came from the proper source, rather than from a spoofing adversary.

Current solution

	Mechanism	Issue
Persistence	DNS, HTTP redirect	Neither work if data moves across domains
Availability	CDNs, P2P	Rely on ad hoc and application-specific mechanisms
Authenticity	IPsec, PKI (public key infrastructure)	Focus on the channel, not on content

Solution from DONA

	Handled by	Provided by
Persistence	Names	Names do not refer to location
Authenticity	Names	Easy authentication
Availability	Name resolution	Route-by-name paradigm

DONA Naming

- DONA names are organized around *principles*

  Each datum or service or any other named entity is associated with a principle.

- Names are of the form of P:L

  Where P is the cryptographic hash of the principal's public key and L is a label chosen by the principle to ensure the names are unique

- To ensure **persistence**

  Names do not refer to location such that data can be hosted anywhere

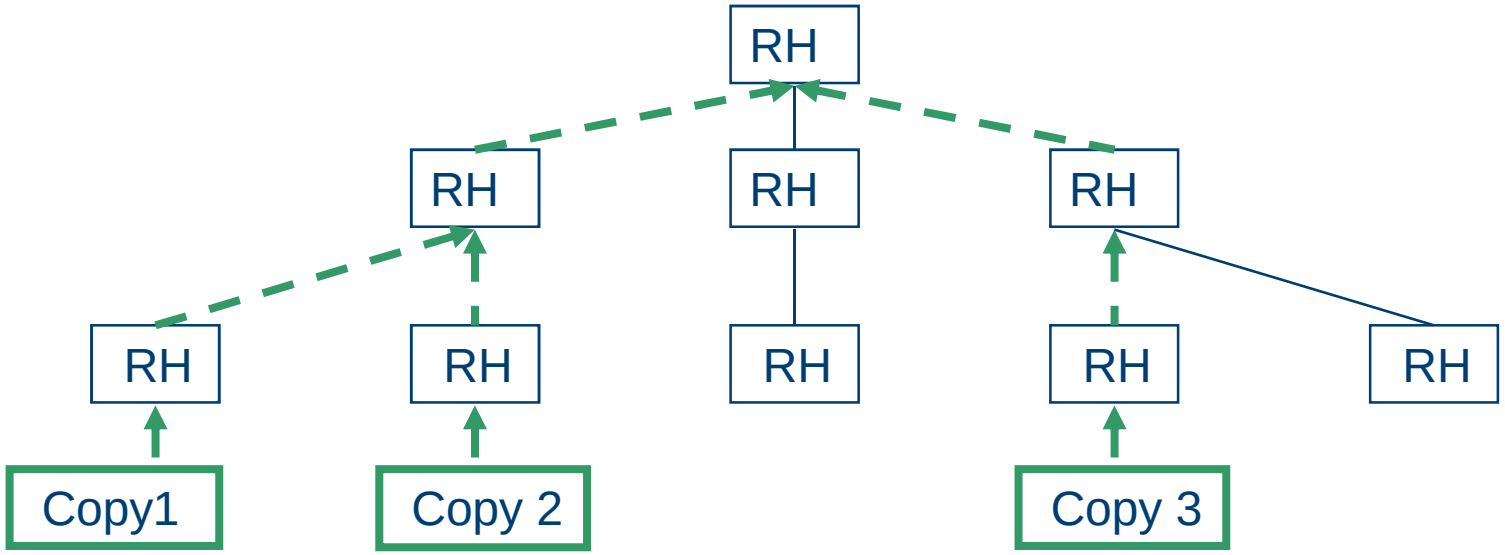
- To ensure **authentication**

  Client who receives data packet in the form of <data, public key, signature> can verify the data by checking the public key hashes to P and this key also generates the signature.

DONA Name Resolution

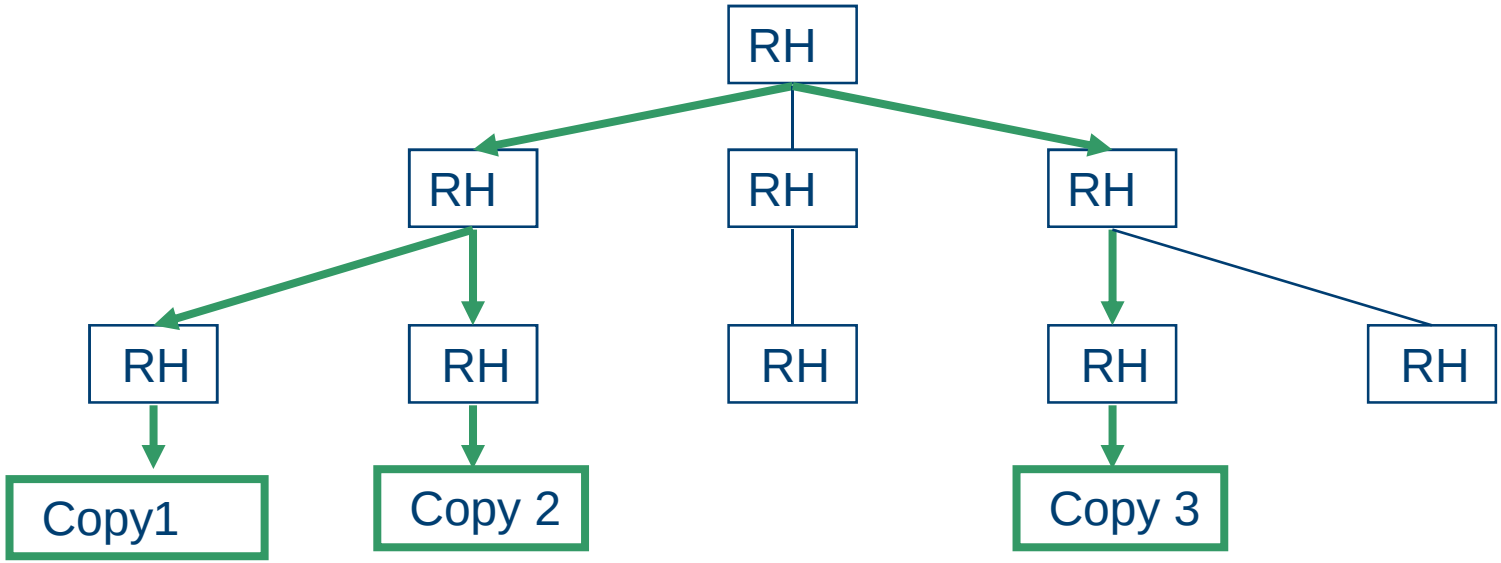
- Route-by-name paradigm for name resolution
  - New class of network entities called RH (resolution handler)
- Operations:
  - FIND(P:L) and REGISTER(P:L)
  - FIND(P:L) locate the object named P:L
  - REGISTER messages set up the state necessary for the RHs to route FINDs effectively
- Process:
  - Find (P:L) request is routed by RHs to a nearby RH where the name P:L is registered

Register Process



- RHs forward register commands to parents and peers if no such record exists before.

Establish Routing State

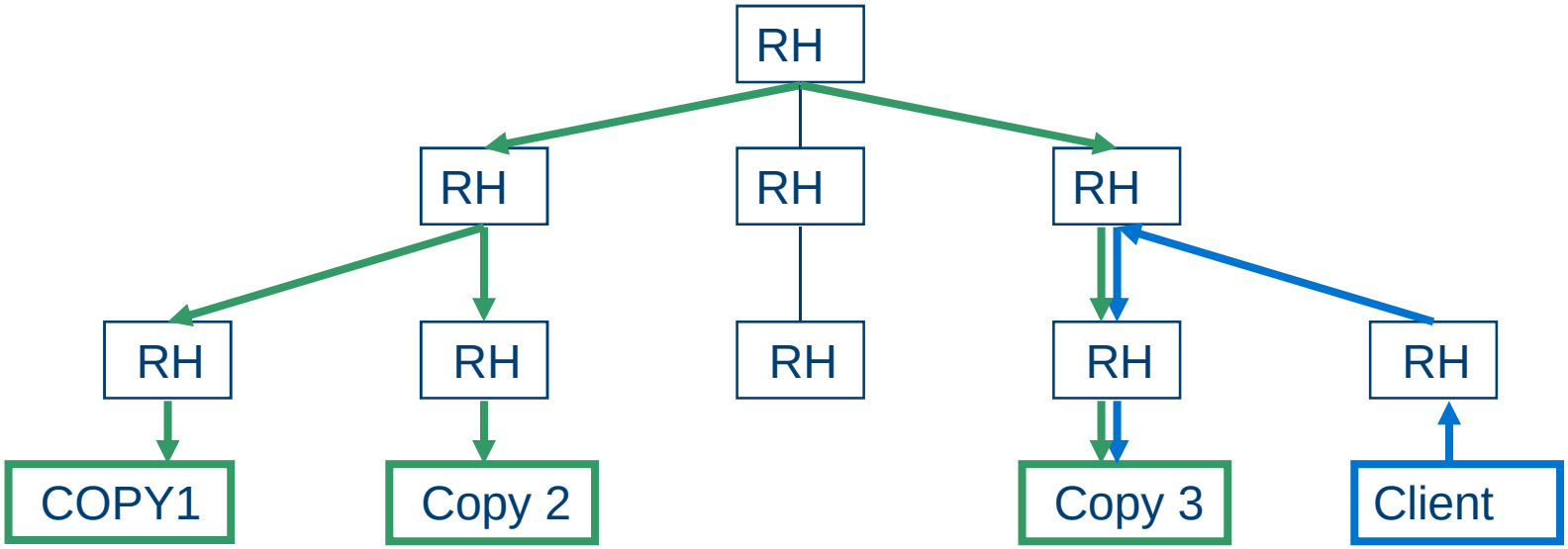


- Arrows represent next-hop entries for registered data

Forwarding FIND(P:L)

- When FIND(P:L) arrives to a RH:
  - If there is an entry in the registration table, the FIND is sent to the next-hop RH
  - If there is no entry, the RH forwards the FIND towards to its parent
- In case of multiple equal choices, the RH uses its local policy to choose among them

FIND Process



- If there's an entry for a data item, follow next-hop
- Otherwise, send to parent
- Standard routing behavior, but at DONA-layer

DONA Basic Functionalities

- Server Selection
- Mobility and Multihoming
- Session Initiation
- Multicast State Establishment

Basic Functionality – Server Selection

Each server (or datacenter) authorized by a principal P to host a service or datum (named P:L) simply registers P:L at their local RH.

DONA routes any FIND(P:L) to the closest such server.

Most basic usage.

Basic Functionality – Mobility and Multihoming

Mobility

A roaming host can first unregister from one location and then re-register at its new location. Subsequent FINDs will be routed to the new location as soon as the new registrations have installed the necessary state.

Multihoming

All interfaces of a multihomed host registers with local RH.
A multihomed domain forwards its REGISTERs to each provider.

FINDs, and thus the resulting data connections, can then make use of multiple paths.

DONA Advanced Functionalities

- Improving content delivery
- Delay tolerant networking
- Access rules and middle boxes

Advanced Functionality – Improving content delivery

- Method #1: Caching
  - An RH should first populate its cache, by changing the source IP address of an incoming FIND packet to be its own before forwarding the FIND to the next-hop RH.
  - When a FIND arrives and there is a cache hit, the RH responds to the FIND's source IP address. Then proceed into a standard application-level exchange.
  - If the RH does not understand the transport or application-level protocol for a particular FIND, it does not provide caching for that request.

Advanced Functionality – Improving content delivery

- Method #2: Subscriptions
  - Often clients would like to subscribe for updates, as in RSS. This can be easily accomplished by adding TTLs to FIND
  - When the server responds to such a TTL'ed FIND, it notes whether and how long it will provide updates to the FIND.
  - When a server updates a piece of content that has a pending TTL'ed FIND, it sends the update to the source of the FIND.

Advanced Functionality – Improving content delivery

- Method #3: Avoiding Misbehaving and Overloaded Servers
  - In general, RHs will route FINDs to nearby copies of the data. However, some of these servers may be misbehaving and stop providing a valid copy of data.
  - To make sure that clients can still access the data, we allow the client to ask that its FIND be routed to a different server instead of the closest one.
  - We also allow overloaded servers to indicate they are overloaded,

Feasibility

- Requirements for a domain's single logical RH:
  - RH only need to keep routing state for data that lie below or equal to it in the AS hierarchy
  - Toughest requirement will be on the Tier-1 providers, whose RHs must keep everything in their registration tables—below is easy

Performance requirement for Tier-1 RH

- Registration processing requirement for a single Tier-1 ISP:
  - 10^{11} names, 42 bytes per entry (40 for the name and 2 for a new-hop RH) = Routing table 4TB ($4* 10^{12}$) to storage
  - Average registration lifetime 2 weeks = 10^{11} /($1.21* 10^{6}$) = 83,000 registration/s a Tier-1 RH must handle
  - Total registration load can be handled by 40 CPUs

Performance requirement for Tier-1 RH

- Forwarding requirements for a single Tier-1 ISP:
  - HTTP request rate for a fully loaded Gbps link is on the order of 20,000 request/s
  - Assume a fully loaded Gbps link can generate 20,000 FIND per second.
  - One FIND packet is 150bytes, 20,000 FINDs= $150*8*20000/10^6$ = 24Mbps only 2.4% of Gbps link
  - If processed in RAM, requires 500 PC, each with 8G RAM
  - If processed in disks, for each incoming Gbps link, need 50 disks

Conclusion:

- Currently, applications are oriented around hosts, addresses and bytes (Socket API)
- However, with DONA, one could use an application interface that is based on the FIND and REGISTER primitives
- With DONA, the applications would revolve around the names of data and services, **not** the address or hostname of their location,
- DONA Advantage:
  - persistence, authentication, availability;
  - Enable application protocols to be oriented around application objects not bytes

Q & A

- Any questions?