Intentional Naming System

Presentation by: Rick Comesa


MIT Laboratory for Computer Science

Intentional Name Resolver

Name Specifier - the intentional name of each resource

AV Pair - Attribute/Value

Late/Early binding

Today

Routers

DNS

Hostname Address

• Mostly static topology & services
  • Hard to maintain
  • Adding new hosts is hard
  • Deploying new services cumbersome
  • Applications cannot learn about network
  • Failures are common!
  • High management cost

Servers

Clients

Resource discovery

• Why is this hard?
  • Dynamic environment (mobility, performance changes, etc.)
  • No pre-configured support, no centralized servers
  • Must be easy to deploy (“ZERO” manual configuration)
  • Heterogeneous services & devices
  • Approach: a new naming system & resolution architecture

Terms

INR - Intentional Name Resolver

Name Specifier - the intentional name of each resource

AV Pair - Attribute/Value

Late/Early binding

INS principles

• Names are intentional, based on attributes
  • Apps know WHAT they want, not WHERE

• INS integrates resolution and forwarding
  • Late binding of names to nodes

• INS resolvers replicate and cooperate
  • Soft-state name exchange protocol with periodic refreshes

• INS resolvers self-configure
  • Form an application-level overlay network
  • Nodes can add/drop from network

Benefits of INS

• Heterogeneous network

• Dynamism
  • Mobility
  • Performance variability
  • Services “come and go”
  • Services may be composed of groups of nodes

• Example applications
  • Location-dependent mobile apps
  • Network of mobile cameras

• Problem: resource discovery

Several Slides Courtesy of: Hari Balakrishnan William Adjie-Winoto Elliot Schwartz Jeremy Lilley

MIT Laboratory for Computer Science
Design goals and principles

- Expressiveness: Names are intentional; apps know what, not where
- Responsiveness: Integrate name resolution and message routing (late binding)
- Robustness: Decentralized, cooperating resolvers with soft-state protocol
- Easy configuration: Name resolvers self-configure into overlay network

Ad hoc configuration

- Static configuration impossible
- DHCP-like configuration undesirable
  - Over wireless, pre-configured subnetworks and broadcasts problematic
- Solution: Distributed, randomized address assignment

Name-specifiers

- Expressive name language (like XML)
- Resolver architecture decoupled from language
- Providers announce descriptive names
- Clients make queries
  - Attribute-value matches
  - Wildcard matches
  - Ranges

Name lookups

- Lookup
  - Tree-matching algorithm
  - AND operations among orthogonal attributes
- Polynomial-time in number of attributes
  - $O(n^d)$ where $n$ is number of attributes and $d$ is the depth

How does it work?

- INR
- DSR
- Application-level routing using intentional names

INS Architecture

- Early binding
- Late binding
- Intentional anycast
- Application-level routing using intentional names

Inter-domain information via DSR protocol

Exchange names as if they were routes

Application-level overlay network formed based on performance
Late binding

- Mapping from name to location can change rapidly
- Overlay routing protocol uses triggered updates
- Resolver performs lookup-and-forward
  - lookup(name) is a route; forward along route
- Two styles of message delivery
  - Anycast - INR returns 1 location (metric based)
  - Multicast

Intentional anycast

- lookup(name) yields all matches
- Resolver selects location based on advertised service-controlled metric
  - E.g., server load
- Tunnels message to selected node
- Application-level vs. IP-level anycast
  - Service-advertised metric is meaningful to the application

Intentional multicast

- Use intentional name as group handle
- Each resolver maintains list of neighbors for a name
- Data forwarded along a spanning tree of the overlay network
  - Shared tree, rather than per-source trees
  - Enables more than just receiver-initiated group communication

Routing Topology

- Routing information
  - Triggered, Periodic updates to other INRs
  - Each update contains:
    - IP Address + name specifiers + port number
    - Port numbers allow early binding
    - Application defined metric
    - Next hop INR + INR’s RTT
    - Announcer ID
      - Unique to each instance of a service

Resolver network

- Resolvers exchange routing information about names
- Multicast messages forwarded via resolvers
- Decentralized construction and maintenance
- Implemented as an “overlay” network over UDP tunnels
  - Not every node needs to be a resolver
  - Too many neighbors causes overload, but need a connected graph
  - Overlay link metric should reflect performance
  - Current implementation builds a spanning tree

Spanning tree algorithm

- Loop-free connectivity
- Construct initial tree; evolve towards optimality
  - Select a destination and send a discover_bottleneck message along current path
Robustness

- Decentralized name resolution and routing in "serverless" fashion
- Names are weakly consistent, like network-layer routes
  - Routing protocol with periodic & triggered updates to exchange names
- Routing state is soft
  - Expires if not updated
  - Robust against service/client failure
  - No need for explicit de-registration

Applications

- Location-dependent mobile applications
  - Floorplan: A map-based navigation tool
  - Camera: A mobile image/video service
  - Load-balancing printer
  - TV & jukebox service
- Sensor computing
- Network-independent “instant messaging”
- Clients encapsulate state in late-binding applications

Routing Protocol Scalability

- vspace = Set of names with common attributes
- Virtual space partitioning: each resolver now handles subset of all vspaces

Scaling issues

- Two potential problems
  - Lookup overhead
  - Routing protocol overhead

- Load-balancing by spawning new INR handles lookup problem
- Virtual space partitioning handles routing protocol problem

- Just spawning new INR is insufficient

Conclusion

- INS is a self-organizing resource discovery system for dynamic, heterogeneous networks
- Expressiveness: names that convey intent
- Application-layer overlay network allows flexible network application development
- Responsiveness: late binding by integrating resolution and routing
- Robustness: soft-state name dissemination with periodic refreshes
- Configuration: resolvers self-configure into an overlay network

Questions?