ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Information-Centric Networks

Section # 7.1: Evolved Addressing & Forwarding

Instructor: George Xylomenos

Department: Informatics

European Union European Social Fund

OPERATIONAL PROGRAMME EDUCATION AND LIFELONG LEARNING investing in knowledge society

MINISTRY OF EDUCATION, LIFELONG LEARNING AND RELIGIOUS AFFAIRS MANAGING AUTHORITY

Co- financed by Greece and the European Union

NSRF 2007-2013 programme for development EUROPEAN SOCIAL FUND

Funding

- These educational materials have been developed as part of the instructors educational tasks.
- The **"Athens University of Economics and Business Open Courses"** project only funded the reformatting of these educational materials.
- The project is being implemented as part of the Operational Program "Instruction and Lifelong Learning" and is co-financed by the European Union (European Social Fund) and national funds.

Licencing

Week 7 / Paper 1

- Internet Indirection Infrastructure
  - Ion Stoica, Daniel Adkins, Shelley Zhuang, Scott Shenker, Sonesh Surana
  - IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 12, NO. 2, APRIL 2004
- Main point
  - Multicast, anycast and mobility are hard on the Internet
  - i3 is an overlay-based alternative
    - Packets are associated with identifiers
    - Receivers use identifiers to complete delivery
    - Sending is decoupled from receiving

Introduction

- With IP the sender knows the address of the receiver
  - Not so with multicast, anycast and mobile hosts
  - A level of indirection is needed (e.g. group addresses)
  - Implementation faces technical and deployment issues
- Many overlay solutions aim to solve this problem
  - Each solution targets a single issue (e.g. multicast)
  - i3 is an attempt to provide a generic solution
  - Based on a general overlay service
  - Specific solutions are built on top of i3
    - Multicast, anycast mobility

i3 overview

- Service model
  - Sources send packets to a logical identifier
  - Receivers express interest to that identifier
    - Best-effort delivery (as usual on the Internet)
  - Slightly different than IP multicast
    - In IP multicast the infrastructure controls the routing
    - In i3 receivers can control packet routing
- Rendezvous-based communication
  - Packets are (id, data) pairs which can be sent
  - Triggers are (id, address) pairs which can be inserted/removed
  - Inexact matching with threshold *k*
    - An id matches a trigger for at least *k* prefix bits
    - There is no longer match with another trigger

i3 overview

- Overview of the design
  - i3 uses an overlay which stores triggers and forwards packets
  - At any time a unique node is responsible for an identifier
    - Triggers for this id are stored there
    - Packets with this id are forwarded there
  - Inexact matching with k bits
    - All id's matching at k bits are handled by the same server
  - End-hosts need to know only a single i3 node
    - That node forwards their requests
  - End-hosts periodically refresh their triggers

i3 overview

- Communication primitives
  - Mobility: a node changes its address
    - The node updates its triggers to point at the new address
    - It is possible to choose triggers handled by a nearby i3 server
  - Multicast: many nodes receive the same packets
    - All group members insert triggers for the same id
    - The sender sees no difference with unicast
  - Anycast: one node of a group receives each packet
    - Group members insert k-matching triggers
    - The sender uses a k-matching identifier
    - The member with the longest matching prefix gets the packet

i3 overview

- Stack of identifiers
  - An identifier stack is a list of identifiers or addresses
  - Packets and triggers may contain stacks
    - Packets are sent to a series of identifiers
    - Packets are forwarded to a series of identifiers
  - Say that a packet contains a stack (id1, id2, id3)
    - i3 tries to match id1, then id2, then id3
    - If id1 does not match any triggers, it is removed
    - Say that id2 matches a trigger with stack (x, y)
    - The packet is forwarded to (x, y, id3)
    - If x is an IP address, the packet is sent there with a tag of (y, id3)
    - The receiver at x may then send a new packet to (y, id3)
    - If packet matches many triggers, it is duplicated for each one

Using i3

- Service composition
  - Example: data transformation for a mobile service
  - Sender stack: first id for the transformer, second for the recipient
- Heterogeneous multicast
  - Example: video to receivers with different codecs
  - Trigger stack: first id for the transcoder, second for the recipient
- Server selection
  - Example: load balancing between web servers
  - Use k-matching prefixes and random trailing bits
    - Each server inserts one or more triggers, depending on capacity
- Large scale multicast
  - Triggers point to servers that further duplicate packets

Design and performance issues

- Overlay properties
  - We need robustness, scalability, efficiency, stability
  - The prototype uses Chord
  - Nodes only use the first k bits for addresses
    - All k-matching prefixes are stored in the same node
- Public and private triggers
  - Distinction made only at the application layer
  - Public triggers are long lived (e.g. web server addresses)
    - Can be produced by hashing the target’s name
  - Private triggers are short lived (e.g. a session identifier)
    - Example: a client chooses a private identifier
    - The client sends the identifier to a server and inserts a trigger for it
    - The server returns its own private identifier and also inserts a trigger
    - A temporary channel now exists between client and server

Design and performance issues

- Robustness
  - Only soft state is kept: triggers must be refreshed!
  - What can we do while a trigger is lost?
  - Use a backup trigger and ask senders to use an id stack
    - If the first trigger is lost, the backup catches the packet
  - Use replication at the overlay level for the triggers themselves
- Routing efficiency
  - Overlay routing is less efficient than regular routing
  - Sender’s should cache i3 server addresses as much as possible
    - If a packet reaches an intermediate server a flag is set
    - The final server returns its address to the sender if the flag is set
  - Avoiding triangle routing by using nearby servers for triggers
    - Receivers can sample the id space to find nearby servers

Design and performance issues

- Security
  - Malicious triggers can wreak havoc into the system
  - Constrained triggers: in (x, y) x constrains y of vice-versa
  - Pushback: dropped packets lead to trigger removal
  - Challenges: trigger insertion requires responding to a challenge
- Other issues
  - Avoiding hot spots: push triggers to other servers
  - Self-organization: achieved by Chord
  - Scalability: triggers are inserted per flow, but only at one node
  - Deployment: Chord can start with a single node and expand
  - Legacy applications: a proxy can be used to insert the id's
  - Anonymity: eavesdropping does not reveal both endpoints

Simulation results

- Evaluation of latency stretch compared to IP
  - Chord simulator with 5000 node topologies
  - **Either power-law random graph or GT-ITM topologies**
- End-to-end latency
  - Assumes senders cache the node responsible for a trigger
  - Also assumes that triggers use sampling to select nearby servers
  - With 16-32 samples stretch stabilizes around 1.5
- Proximity routing from sender to i3 server
  - How much does it cost to send the first message?
  - Closest finger replica: maintain the successors of each finger
  - Closest finger set: evaluate multiple fingers and keep closest
  - With 10 replicas both strategies perform similarly
  - Stretch is 5-7 with the heuristics, 10-16 without

Simulation results

- Implementation and experiments
  - Bare-bones implementation based on Chord
  - 256 bit identifiers with k=128
  - 48 byte header in all packets and triggers
    - Packets can also carry up to 4 identifiers in a stack
  - Triggers updated every 30 seconds
- Performance
  - Trigger insertion: 12.5 μs (table lookup and memory allocation)
  - Data packet forwarding: 15-25 μs, linear on packet size
    - Lookup matching triggers and forward, only unicast
  - i3 routing: 20-30 μs, linear on packet size
    - Lookup finger table and forward, finger table is a list (slow!)
  - Throughput: 53-260 Mbps, better with larger packets

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

End of Section # 7.1

Course: Information-Centric Networks, **Section # 7.1: Evolved Addressing & Forwarding**

Instructor: George Xylomenos, **Department:** Informatics