The PROXIDOR Service

draft-akonjang-alto-proxidor-00.txt

S. Previdi (sprevidi@cisco.com)
O. Akonjang (obi@net.t-labs.tu-berlin.de)
A. Feldmann (anja@net.t-labs.tu-berlin.de)
B. Davie (bdavie@cisco.com)
D. Saucez (damien.saucez@uclouvain.be)

IETF74 - San Francisco, March 2009 - ALTO WG

Proxidor Service - Introduction

- Proxidor is a signaling protocol serving different implementations of “localization/location-based” services
  - Draft describes high-level architecture
  - Detailed protocol specification will follow

- Merger of three existing proposals, easily extensible to others
  - Proximity (Cisco),
  - Oracle (DT-Labs),
  - IDIPS (UCL Belgium)

Proxidor Service - Introduction

- Address ALTO requirements in terms of p2p localization signaling
  - Other applications possible, e.g., currently used in some content networking applications

- Client/Server Model
  - Client: any application element with embedded Proxidor API
    - Example: p2p client, CDN client, CDN server, ...
  - Server: interfacing application layer with routing layer
    - Operated by SPs
  - A Proxidor server can also act as a Proxidor client

Proxidor Service - Messaging

- Proxidor Protocol
  - TLV based encoding
    - Allows backward-compatible protocol extensions
    - Simple, Flexible, Scalable
    - Routing protocols successfully leveraged TLV encoding in the last decades
  - Not bound to a specific transport layer
    - supports TCP, UDP, HTTP, SOAP, ...

- Messaging (high level)
  - Query (PxQ Message)
    - Example: unsorted list of IP addresses/Prefixes
  - Response (PxR Message)
    - Example: ranked list of IP addresses
    - Example: re-direct to another server
    - Example: combination of two above

Proxidor Service - Messaging

- Message content (high level)
  - PSL: Proxidor Source List
    - Reference for ranking computation (IP Address, Prefix, AS, ...)
    - Single or multiple
    - Can be implicit (i.e.: taken from IP Src address)
  - PTL: Proxidor Target List
    - List of IP Address, Prefix, AS, ...

Proxidor Service - Ranking System

- Ranking System
  - Rank IP identifiers: IP addresses, Subnets, ASs, ...
  - Proxidor Server ranks PTL based topology distance from PSL
  - Rank based on
    - Routing/Topology information (including geographical)
    - SP defined Policies
    - Resources utilization data (non real-time)
- Topology/Infrastructure Information sources
  - Routing Protocols/Database
  - Extensible to backbone/infrastructure resource-state information
  - Extensible to application networking state information
- SP will not publish any topology information
- Ranking algorithm details are not going to be standardized

Proxidor Service - Example:
P2P Neighbor/Peer selection

- P2P Client sends unsorted list of potential neighbors or potential peers for content exchange
  - PSA: client IP address (or inferred by IP Src address)
  - PTL: unsorted list of IP addresses

- Server rank list of IP addresses based on
  - Topology information (routing layer)
  - Transit/backbone resources utilization
  - Data traffic direction (when available)
  - SP policies

- Server replies with ranked PTL

- P2P Client may override rank position based on other criteria

Proxidor Service - Summary

- Proxidor protocol serves different ALTO-like applications
- Extensible to include “location-based” services without fundamental changes
- Lightweight, simple protocol on top of transport layer
- TLV based: efficient, scalable, application agnostic
- Draft describes architecture. Second draft with protocol details will follow
- Proximity, Oracle, Idips are first implementations leveraging Proxidor protocol

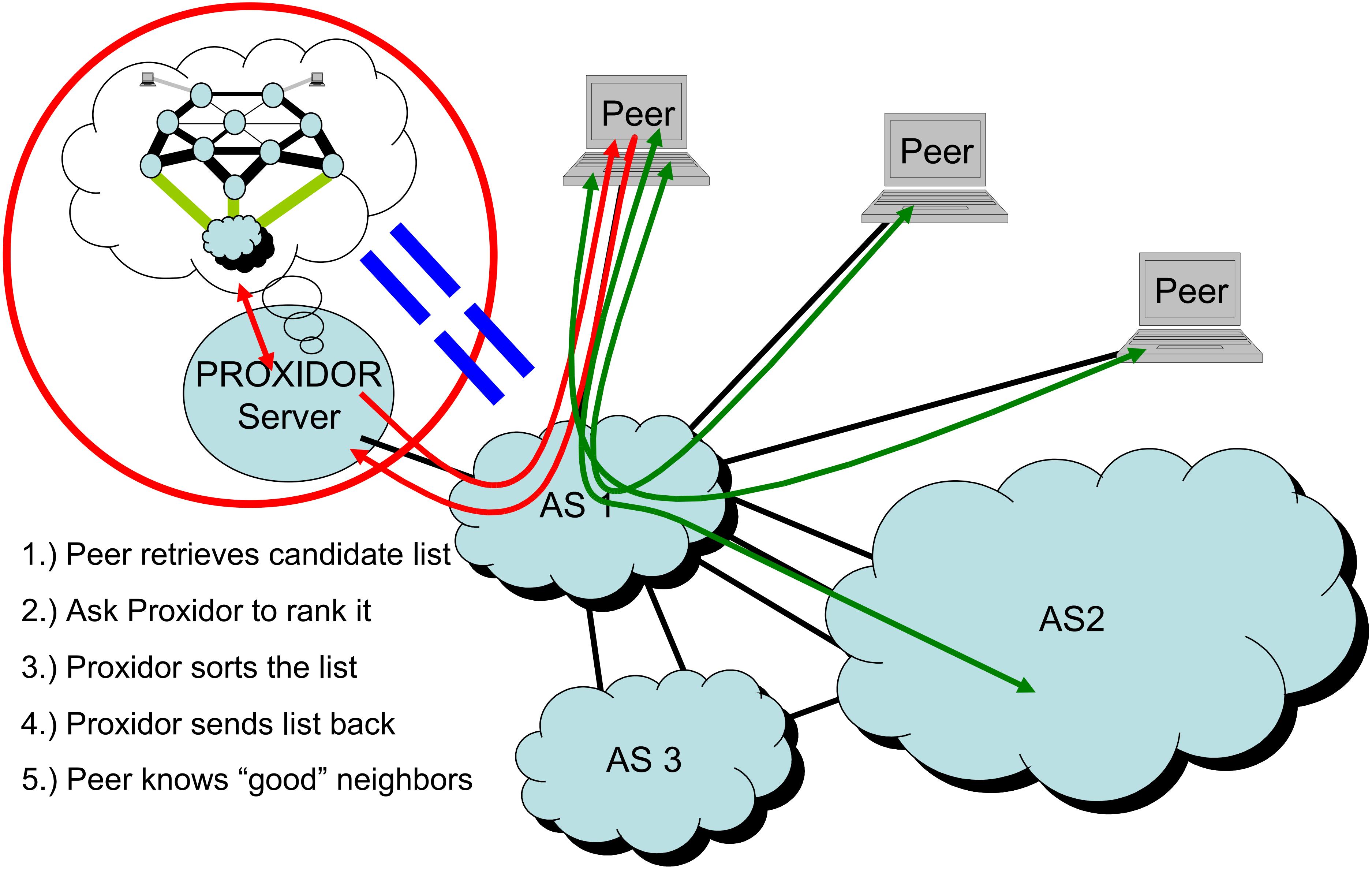
Proxidor Service - Next Steps ?

- Different location-based implementations exist
- Most protocol requirements are common, but not all
  - A protocol specification including all requirements from all existing implementations may be difficult/challenging
- Split and simplify the work
  - Main protocol specification: headers, main TLVs, state machinery, operations
  - Protocol extensions: specific (set of) TLVs in order to accommodate implementations
- Large consensus is required for main specification
- Make protocol extensions optional
  - TLV encoding allows backward compatibility and interoperability
  - E.g.: unknown/unsupported (sub)TLV is silently ignored

Implementation of a PROXIDOR Use Case

Ingmar Poese
Obi Akonjang
Anja Feldmann
Georgios Smaragadakis
{ipoese,obi,anja,george}@net.t-labs.tu-berlin.de

PROXIDOR use case: P2P Oracle



1.) Peer retrieves candidate list

2.) Ask Proxidor to rank it

3.) Proxidor sorts the list

4.) Proxidor sends list back

5.) Peer knows “good” neighbors

PROXIDOR server features

- Scalable architecture
  - Multithreaded design
  - Two levels
    - Internal AS topology
    - External AS topology
  - Relies on caching
- Request handling
  - Via UDP
- Configuration
  - Via files with dynamic updates
- Interactive console

Prototype implementation

- Language: C++
- System: Linux 2.6 Kernel
- Multithreading: pthreads
- Synchronization: pthread mutexes
- Communication: standard Linux sockets
- Sorting: stlib qsort
- Licence: GPL

Initial performance study

- AS: 10 routers with 3 external connections
- Number of external routes: 67000
- Test system: Dual Intel(R) Xeon(R) (E5410 @ 2.33GHz) with 8 Gbyte RAM
- Ubuntu 8.04 Server (32-bit)
- 1 client querying the server at max speed via UDP
- 100 IPs for ranking for each query

Initial performance results

Threads	RAM	CPU-time	Queries per second
1	7.1MB	90.81s	10,500
2	7.1MB	182.80s	19,250
4	7.2MB	358.78s	27,000