Individual Proposals

IETF-70 P2PSIP

5 minutes each

draft-bryan-p2psip-reload-02

C. Jennings, B. Lowekamp, E. Rescorla, J. Rosenberg

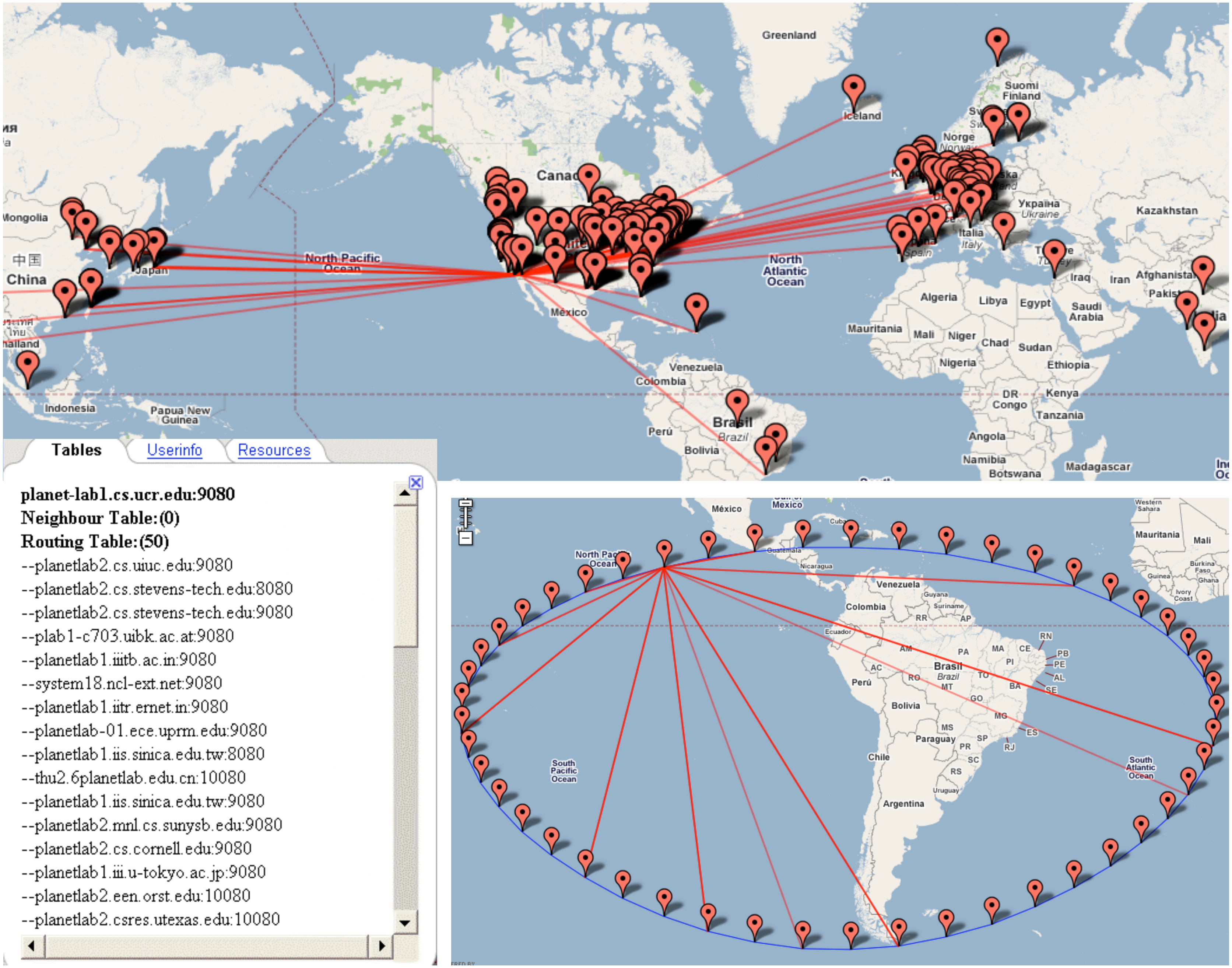
- merge of RELOAD and ASP proposals
- binary protocol
  - fixed fields where possible, TLV where flexible types are needed
- Certificates for Peers and for Users/Resources
  - Proposed enrollment mechanism
  - Also PSK technique
- TLS/TCP or DTLS/UDP with fragmentation

reload-02 Routing

- VIA/Route Log headers with IDs
  - PeerIDs or opaque local ids for clients/compression
- Use ICE
  - gather candidates over time
  - CONNECT with ICE
  - TUNNEL to communicate across overlay
- Supports recursive symmetric/asymmetric and iterative routing
  - Some discussion of pros and cons, more work needed
- Service discovery for STUN/TURN servers
  - assumes predictable percentage of candidates

Peer-to-Peer Protocol (P2PP)

- Structured and unstructured
- Node and data model
  - peers, [clients], [enrollment | diagnostic | other server]
- Improved hop-by-hop reliability model
- Security
  - peer-ID assignment, routing (TLS, DTLS), storage (signature)
- NAT traversal (peer protocol, SIP, media)
- Implementation
  - 500+/100+ nodes Kademlia/Bamboo overlay on ~160 planet lab machines
  - Mobile phones
  - Source code release soon



Tables
Userinfo
Resources
planet-lab1.cs.ucr.edu:9080
Neighbour Table:(0)
Routing Table:(50)
--planetlab2.cs.uiuc.edu:9080
--planetlab2.cs.stevens-tech.edu:8080
--planetlab2.cs.stevens-tech.edu:9080
--plab1-c703.uibk.ac.at:9080
--planetlab1.iiitb.ac.in:9080
--system18.ncl-ext.net:9080
--planetlab1.iitr.ernet.in:9080
--planetlab-01.ece.uprm.edu:9080
--planetlab1.iis.sinica.edu.tw:8080
--thu2.6planetlab.edu.cn:10080
--planetlab1.iis.sinica.edu.tw:9080
--planetlab2.mnl.cs.sunysb.edu:9080
--planetlab2.cs.cornell.edu:9080
--planetlab1.iii.u-tokyo.ac.jp:9080
--planetlab2.een.orst.edu:10080
--planetlab2.csres.utexas.edu:10080
Greenland
Iceland
North Pacific Ocean
North Atlantic Ocean
China
Japan
Mongolia
Indonesia
México
Venezuela
Colombia
Brasil
Perú
Bolivia
Algeria
Libya
Egypt
Mauritania
Mali
Niger
Chad
Sudan
Nigeria
Ethiopia
Kenya
Tanzania
Angola
Namibia
Botswana
Madagascar
Kazakhstan
Iran
Iraq
Afghanistan
Saudi Arabia
South Pacific Ocean
South Atlantic Ocean
Chile
Argentina
Paraguay

HIP-HOP and ID-LOC

Philip Matthews

Eric Cooper

Alan Johnston

Avaya

HIP-HOP and ID-LOC

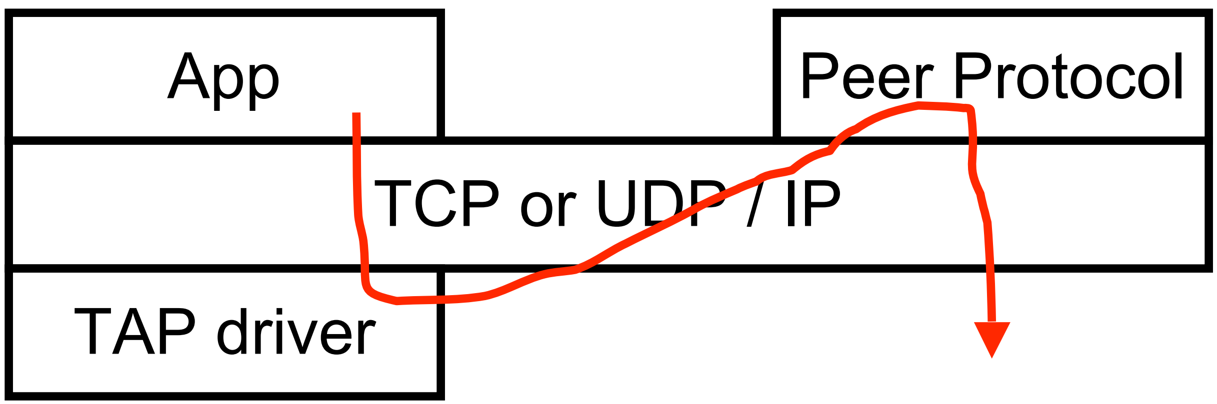
- No new revision of HIP-HOP draft this cycle.
  - Some open issues still being worked on.
- New ID-LOC draft focuses on HIP idea with “biggest bang” for P2PSIP.
  - “ID / Locator split” concept

ID-LOC

- Goals:
  - Make existing apps work in P2P overlays, often without change
  - Transparently handle NAT Traversal
  - Transparently handle Mobility
- Key Ideas:
  - Apps use special IP addresses to identify remote peers
  - Special addresses then translated to real addresses below transport layer
  - Dynamically establish a connection, then send packet on connection

Implementation

- Use VPN techniques
- Packets intercepted by TAP driver and sent to Peer Protocol, which makes necessary adjustments and resends them.



XPP/PCAN Status

- Implemented/specified GRUU and outbound (sortof)
- Implemented/specified replication
- Implemented/specified stabilization for CAN
  - “don't react, stabilize!”
- Implemented STUN and started testing with NATs
  - a nightmare!
  - implemented address changes adaptation
- Still no ICE :-(
  - but... what happens if your 20+ mappings change at once? (UML does that all the time)

Utilizing HIP for P2PSIP (WITH-HIP)

`draft-hautakorpi-p2psip-with-hip-01.txt`

Jani.Hautakorpi@ericsson.com
Gonzalo.Camarillo@ericsson.com
Joakim.Koskela@hiit.fi

Overview

- WITH-HIP **is not** a Peer Protocol proposal
- WITH-HIP can be used **with** Peer Protocols, such as RELOAD and P2PP, for example
- WITH-HIP defines how **unmodified** HIP can be utilized in P2PSIP networks

Why use WITH-HIP?

- HIP provides the following functions:
  - Setup and maintenance of connections between peers
  - Mobility & multi-homing
  - Cryptographic host identities
  - NAT traversal below the application layer

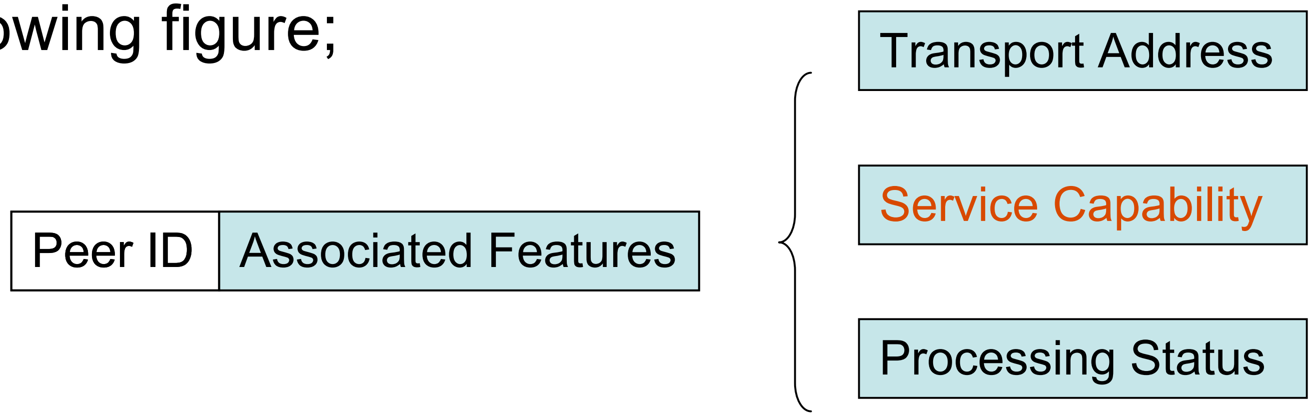
Service Extensible Peer Protocol (SEP)
draft-jiang-p2psip-sep-00.txt

jiang.x.f@huawei.com

hwzheng@huawei.com

Service Advertisement

- The Advertisement of the service capability
  - Each peer encodes its service capabilities;
  - Each peer advertises the info by using overlay maintenance mechanism;
- The peers' routing states are often organized like the following figure;



- So the info about service peers has already been advertised through the overlay;

Service Discovery

- Discovery Method
  - The peer in need of a specific service indicates its desire in the request
  - The intermediate peers and the destination peer collect the info about the service peers;
  - The source peer MAY get the desirable information in the response;
- SEP defines a new message: LookUpServicePeer
  - It also could be done in a piggyback mode;

NAT Traversal for Semi-Recursive

- Semi-Recursive mode
  - Request is routed hop-by-hop through the overlay;
  - Response goes back directly to the source peer;
- Requirements for relaying peers
  - MUST be accessed directly by the destination peer;
  - MUST know how to relay the response to the source peer in the presence of the NATs;
- The choice for the Relaying Peers
  - Neighbor peers with public address;
  - Any peer with public address;
  - Etc;

P2PSIP Client Protocol

draft-zheng-p2psip-client-protocol-00

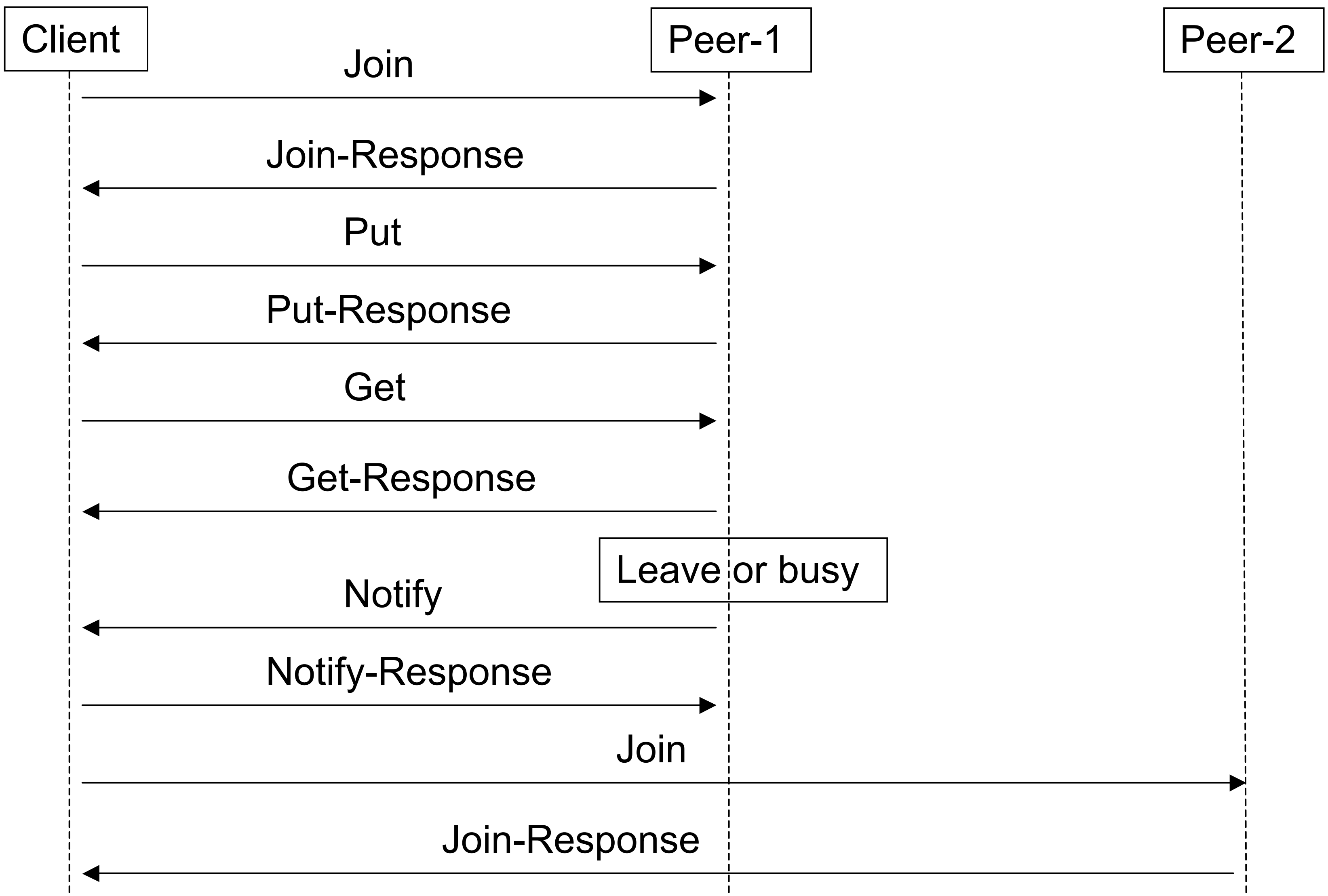
jiang.x.f@huawei.com

hwzheng@huawei.com

What is Client protocol?

- A logical subset of Peer protocol
- Provide data storage and retrieval functions thru client's peer (e.g. GET/PUT/Remove)
- Provide connection control function (e.g. Join/Leave)
- Provide overlay service redundancy function (e.g. Notify)

Sample


Client
Peer-1
Peer-2
Join
Join-Response
Put
Put-Response
Get
Get-Response
Leave or busy
Notify
Notify-Response
Join
Join-Response

Peer-to-Peer Name Service (P2PNS)

draft-baumgart-p2psip-p2pns-00.txt

Ingmar Baumgart
Institute of Telematics, Universität Karlsruhe

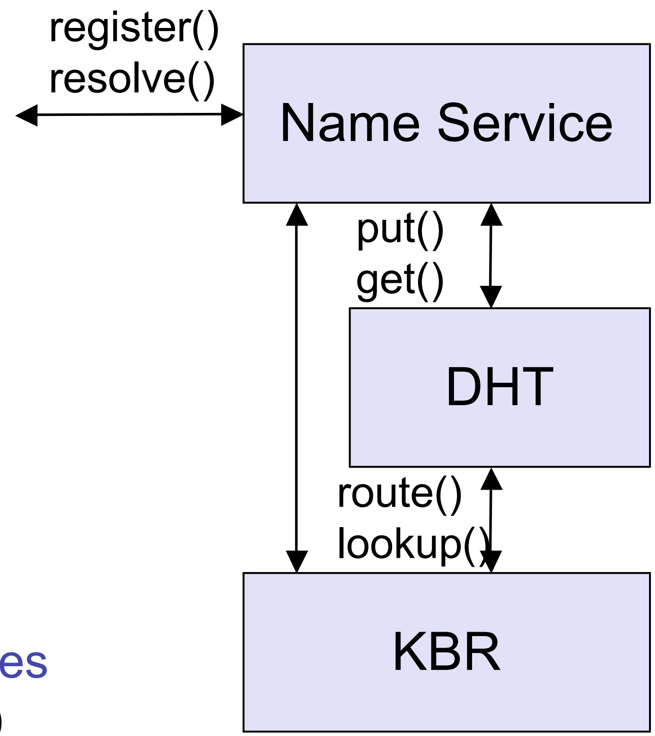
IETF 70, Vancouver

Flexibility

- Distributed name resolution for:
  - P2PSIP, decentralized DNS, HIP, decentralized IM (XMPP)
- Same task in all scenarios:
  - Resolve a P2PName (AoR, Domain Name, HIT) to the current transport address (IP, Port)
- P2PNS XML-RPC Interface:
  - register(P2PName, transport address)
  - resolve(P2PName)

Modular Architecture

- Key Based Routing (KBR)
  - Task: Message routing to nodeIDs
  - route(key, msg)
  - lookup(key)
- Distributed Hash Table (DHT)
  - Task: Data storage
  - put(key, value)
  - get(key)
- Name Service
  - Task: Resolution/Caching of P2PNames
  - register(P2PName, transport address)
  - resolve(P2PName)



→ Modular architecture allows to reuse implementations for different applications (ALM, Filesharing, Gaming,…)

Two-Stage Name Resolution

1.) Resolve AoR → NodeID (DHT layer)

2.) Resolve NodeID → IP (KBR layer)

Motivation:

- Modification of data records on DHT is expensive (due to security mechanisms)
- (AoR, NodeID) binding is static: No modification needed if IP address changes
- IP address changes are efficiently handled on KBR layer

P2PNS Security

- KBR layer:
  - Limit nodeID generation (crypto puzzles or offline CA)
  - Routing over disjoint paths
  - Secure routing table maintenance
- DHT layer:
  - Replication and majority vote
  - Only owner may modify data records (nodeID signature)
    - Prevents identity theft
    - Unique usernames (same key in DHT is only allowed once)
  - Insertion DoS attack prevention