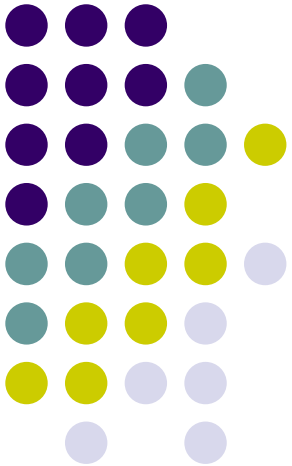
CLOAK Virtual Networking Architecture

Damien Magoni *et al.*
LaBRI – CNRS
University of Bordeaux



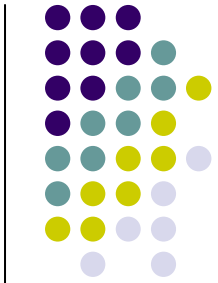
Aims

- What: virtual networking over the Internet
- Which is for:
  - messaging, conferencing, sharing, streaming
- Why:
  - solve issues (scalability, reliability, usability)
  - add features (mobility, flexibility, security, autonomy)

Means

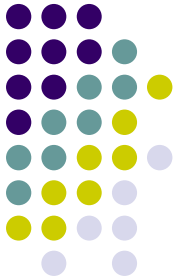
- How:
  - P2P links
  - Overlays
  - Abstract names
  - Distributed Hash Tables
- Where: Internet
- When: asap!

Summary



- Internet issues
- Concepts
- Paradigm and design
- Operations
- Architecture
- Usages
- Performance evaluation
- Pointers

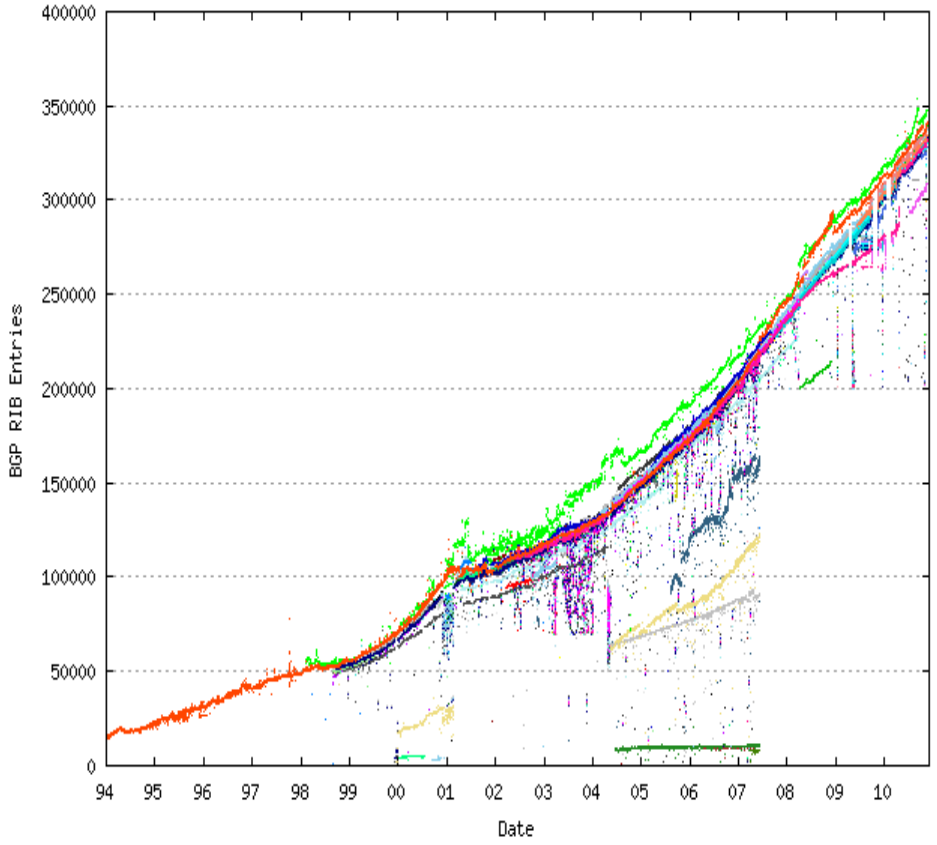
Current issues in the Internet



- Not scalable
  - Addressing space exhaustion for IPv4
  - Routing tables explosion
- Not reliable (in dynamic environments)
  - Limited mobility at network and transport layer
  - No flexibility (transferable connections)
  - No simultaneous mobility/flexibility + security
- Not usable (broken E2E model)
  - Many addressing spaces (v4/v6/NAT)
  - Many middleboxes (NATs, proxies, etc)
  - No user definable namespaces (only official DNS)

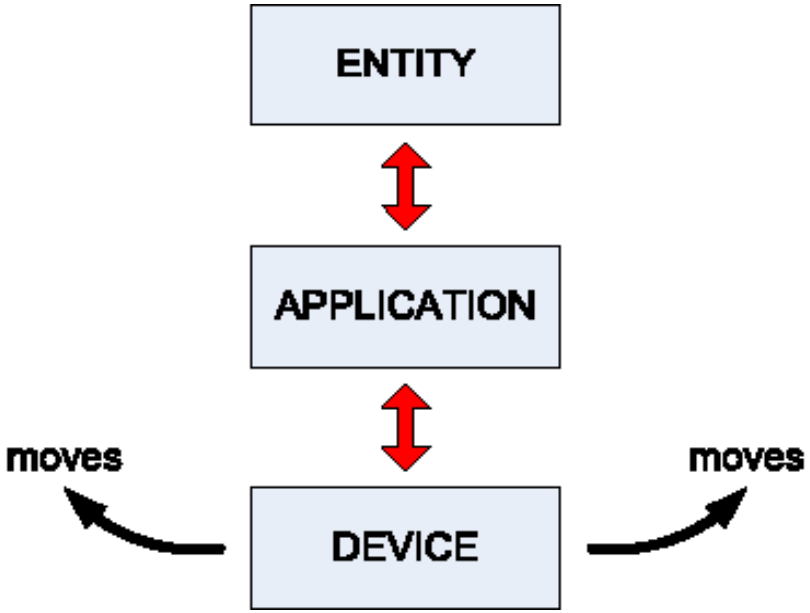
Scalability issues

- IPv4 addressing space exhaustion
- Routing tables explosion (e.g. BGP)
- Current routing protocols are not scalable (OSPF, BGP, AODV, DSR, etc)



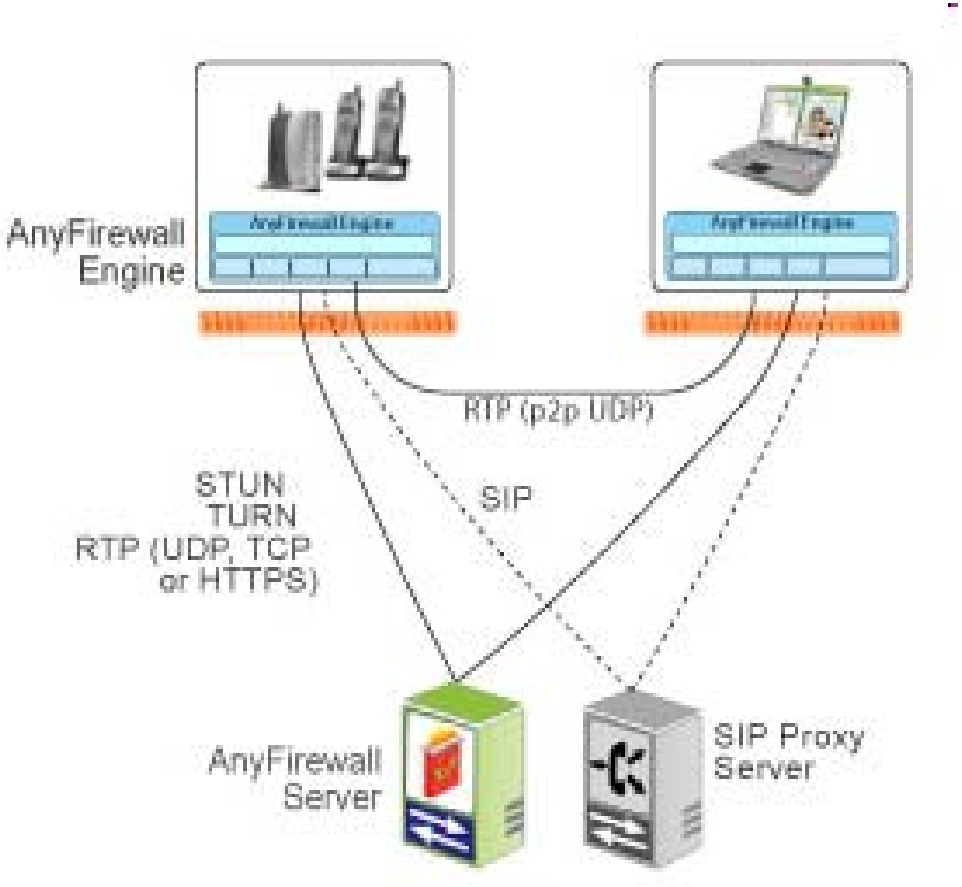
Reliability issues

- Wide device mobility using link layer technologies (WiFi, WiMax, 3G, 4G)
- Limited network and device mobility using network layer protocols (MobileIP)
- Limited transport switching based on ad hoc solutions (Rocks, FT-TCP, TCP-Migrate)



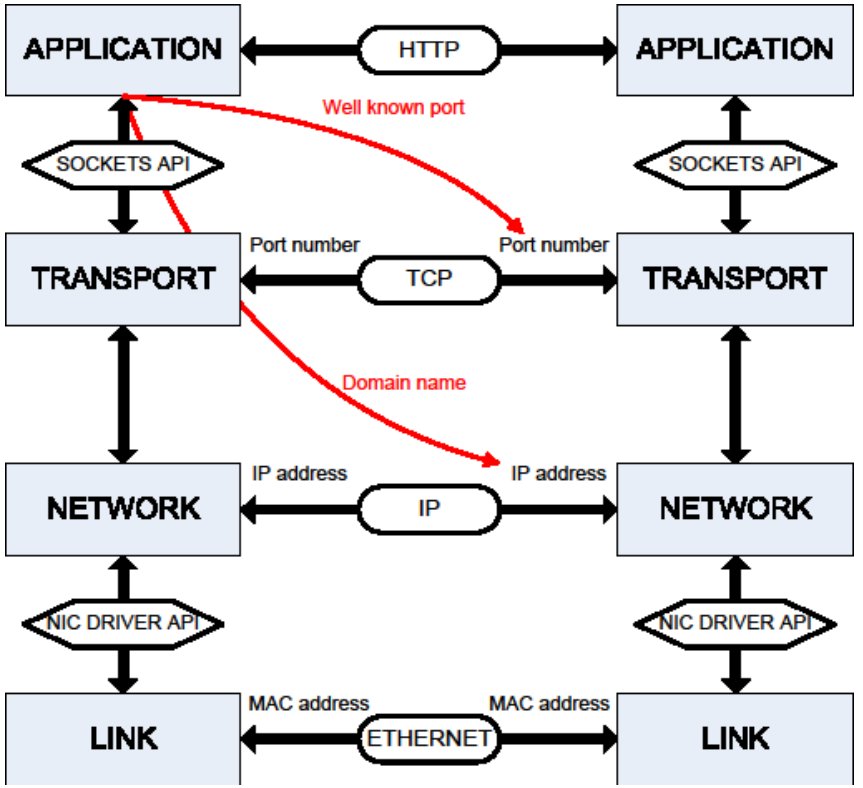
Complexity issues

- IPv4/v6 gateways
- Proxies
  - Web, VoIP, E-mail
- Firewalls
- NATs
  - UPnP (IGDP), ICE (STUN, TURN)

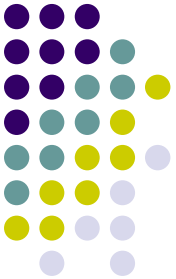


Frozen stack

- Application bound to a location (IP@)
- Application bound to a transport protocol (protocol n°)
- Application bound to an OS managed multiplexing ID (port n°)
- Connection is broken if anything changes



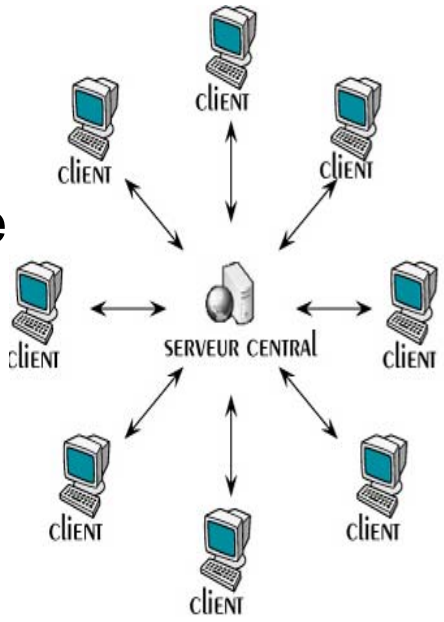
CLOAK applied concepts



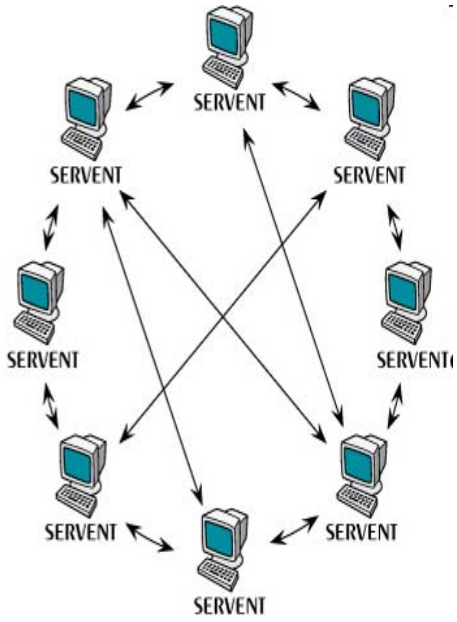
- P2P links built with transport layer connections
  - Dumb network
- Overlays
  - E2E model restored by unique addressing space
- Names
  - Definable name spaces
- DHTs
  - Autonomous storage for communication information

Peer-to-Peer networking

- Device can work as a client and as a server
- Any device can interact with any other if possible
- No role hierarchy
- No constrained topology
- No single point of failure
- No device or link overload



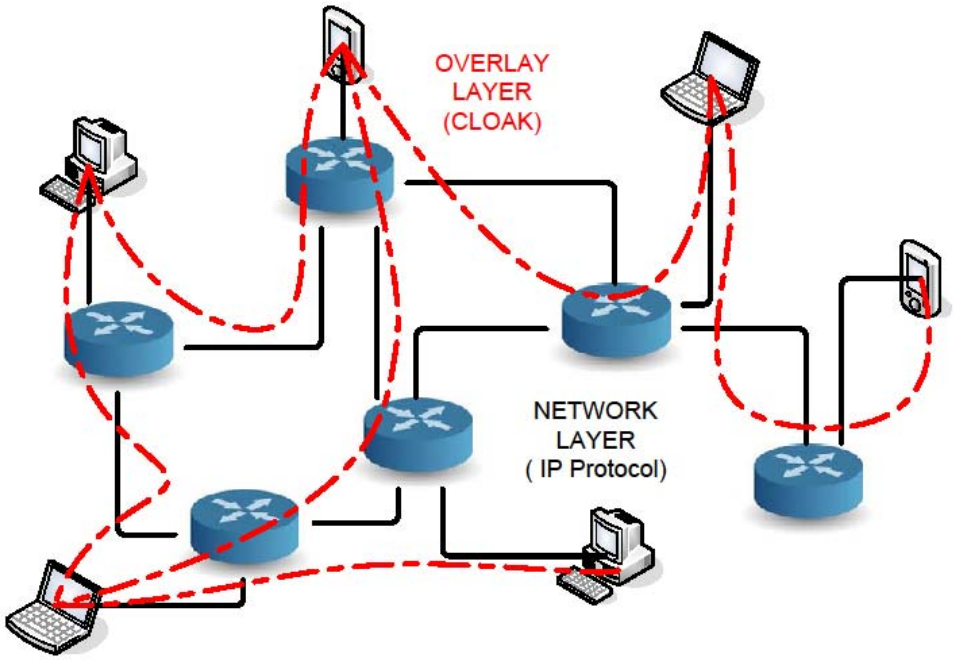
Architecture client-serveur



Architecture pair-à-pair

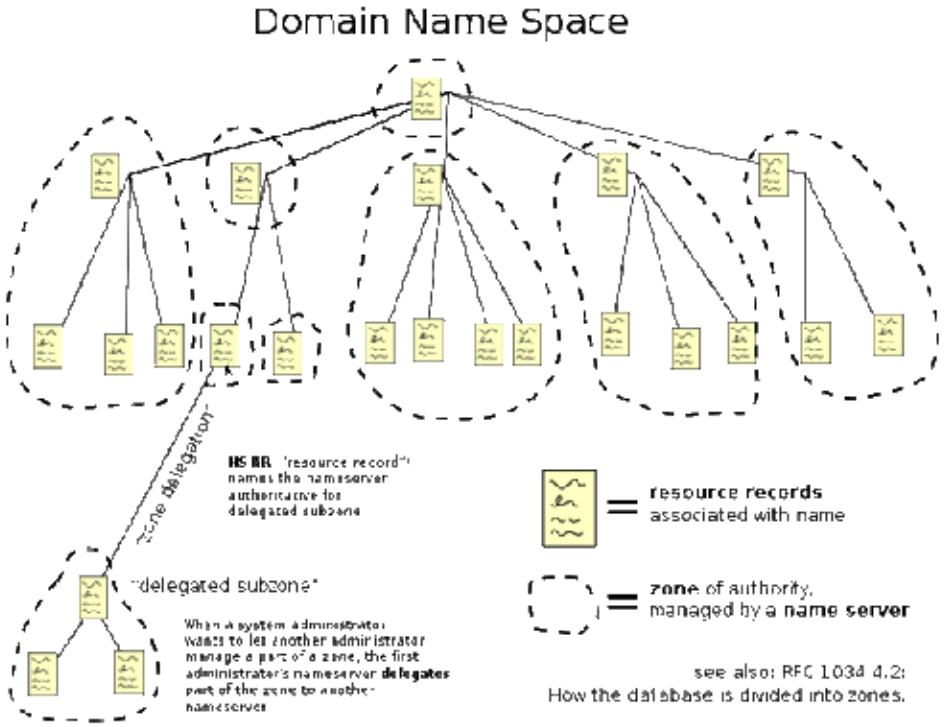
Overlay networking

- Terminal devices connect to some others creating virtual links
- Devices with 2+ links play the role of routers
- Devices form a new virtual network called an *overlay*



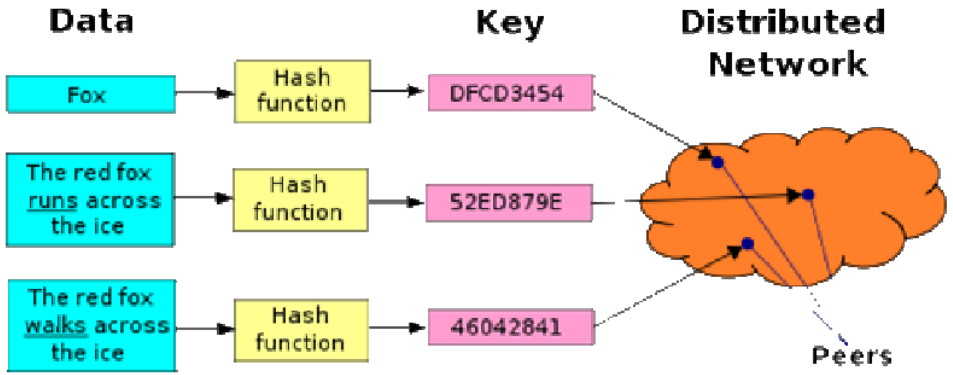
Identifiers & names

- The Internet gives IDs
  - Host names (server.abc.gov)
  - Service names (smtp)
  - E-Mail, IM and VoIP @s (bob@abc.edu)
  - URLs (http://www.abc.com/path/file.html)
- All IDs contain IP@s and PORT n° in disguise!
- The DNS is mostly static
- ID (name) / Locator (@) separation is needed
- Dynamic binding is needed



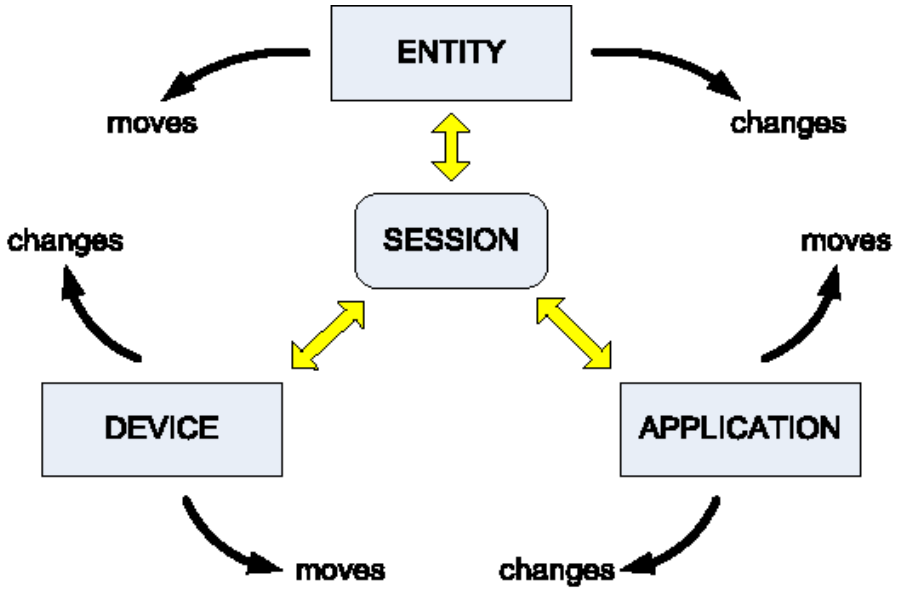
Distributed Hash Tables

- A Hash Table is a directory/map storing (key, value) pairs
- HT split, replicated and located on many devices => DHT
- Queries in log(n)
- Greedy key-based routing
- Consistent hashing

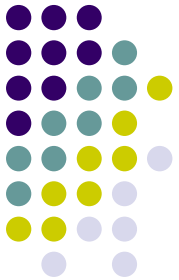


CLOAK paradigm

- A *communication* is movable or transferable across devices and/or entities and/or applications at will and on the fly (when it makes sense)
- A *session* is a container storing all the control information needed during the lifetime of the communication



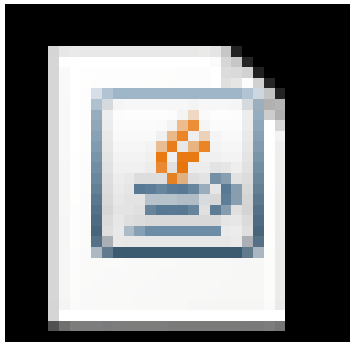
Design



- Addressing (distributed in each peer)
- Routing (local knowledge / compact)
- Naming (for devices and entities)
- Binding (DHT)
  - (device name, overlay @)
  - (entity name, device name)
- Steering (dynamic / on-the-fly binding)

The Poincaré disk model

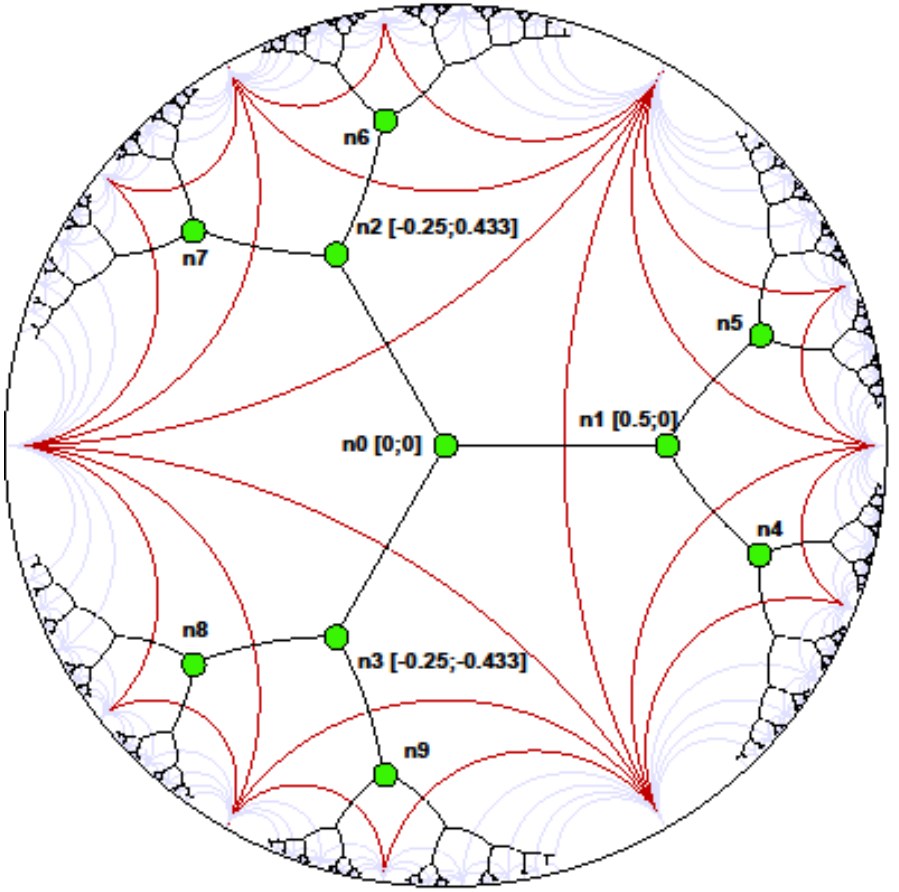
- A model of the hyperbolic plane
- The open unit disk represents the plane
- Points are complex numbers
- Lines are arcs cutting the horizon at right angle
- The unit circle represents points at infinity (horizon)
- Applet (thanks to Don Hatch!)



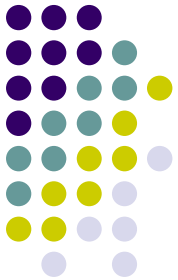
hyperb.jar

Addressing in the unit disk

- Each peer is a node of the tree
- Addresses taken from the regular tree of degree *k*
- *K* is arbitrarily fixed at the overlay creation
- 200M+ addresses when using *double* with precision limited to 10^{-12}



Routing in the Poincaré disk



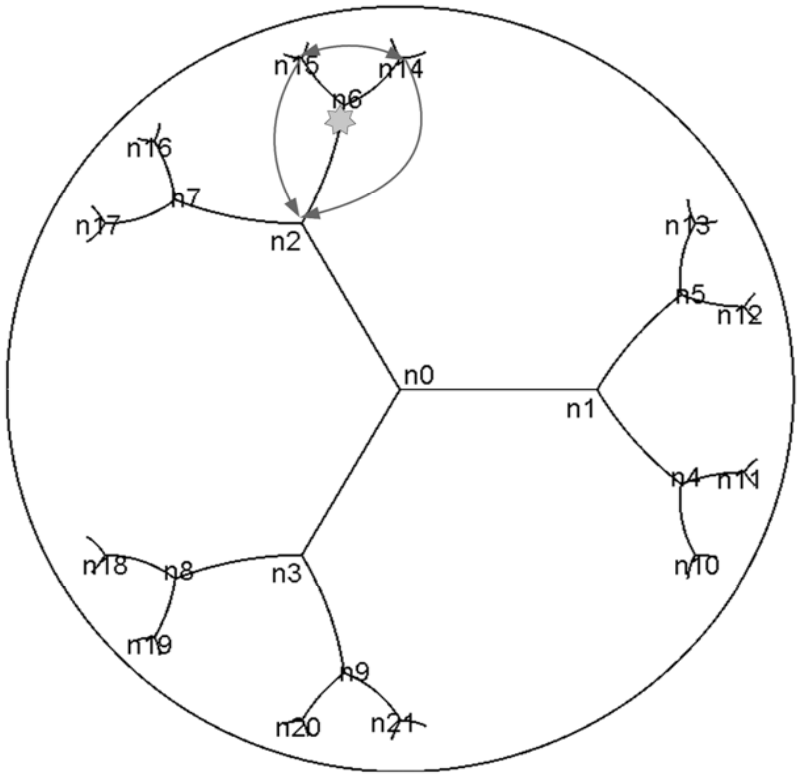
- Greedy routing by using hyperbolic coordinates in the unit disk (represented by a complex number)
- Hyperbolic distance *D(a,b)* between *a* and *b* is given by:

$$D(a,b) = \operatorname{arg\,cosh}\left(1 + \frac{2|a-b|^2}{\left(1-|a|^2\right)\left(1-|b|^2\right)}\right)$$

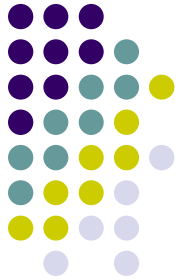
- When a packet to d enters peer p:
  - Compute *D(n,d)* for each neighbor peer *n*
  - Choose the next hop *n* that has the lowest *D*

Dynamic routing

- On peer or link failure
  - Flush @s of descendants (root can't die)
  - Replace the missing and give same @ (must implement @->name in DHT)
- Maintain connectivity and greedy embedding
  - Connect to ascendants
  - Connect to siblings

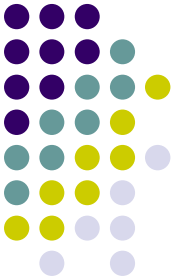


Naming



- Names must be defined by the users of the overlay for entities and devices
  - They must be unique inside the overlay
- Names are stored in the DHT formed by the peers of the overlay
- Group names can be defined
  - For multicast
- Entities' names can be layered (aliases)

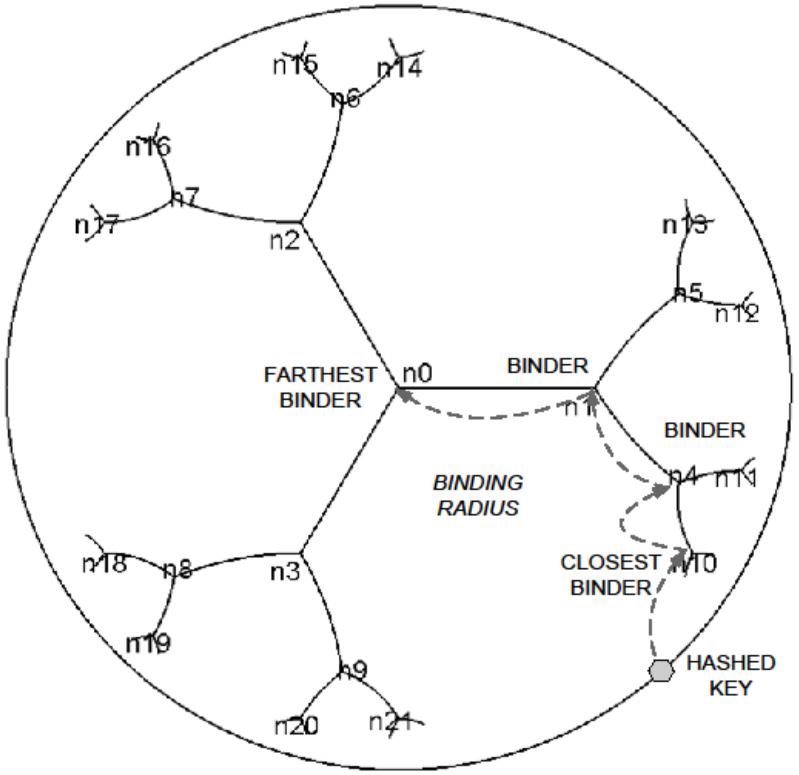
Mapping keys to points



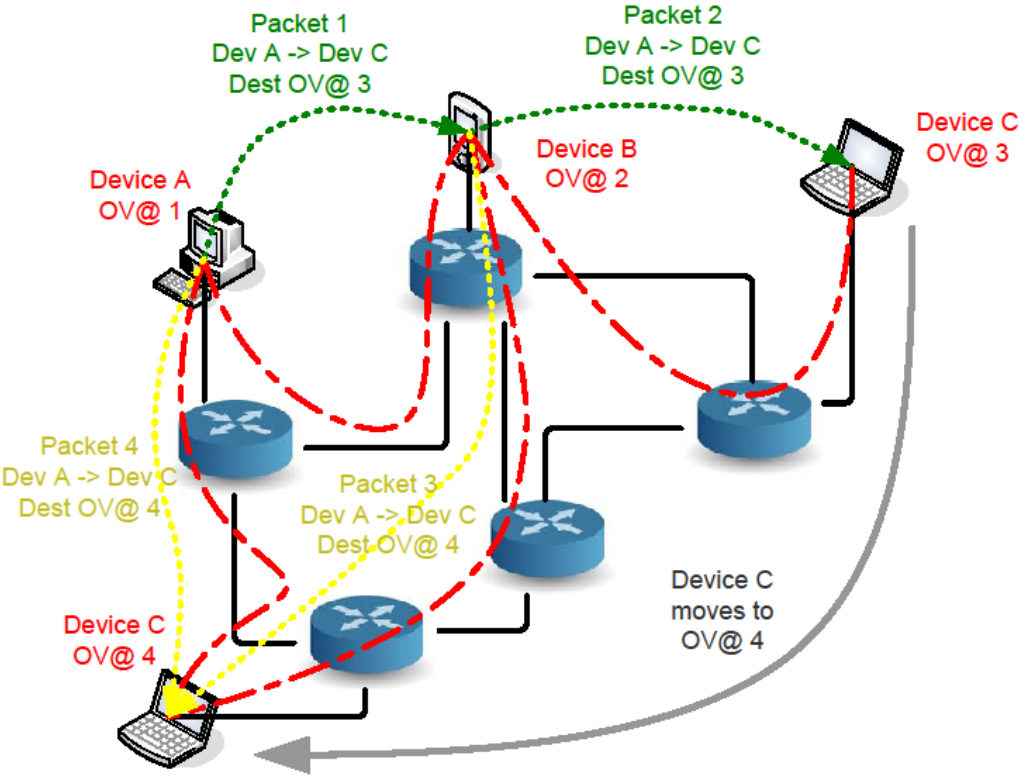
- Pairs can be (device name, overlay @), (entity name, device name), (alias name, entity name), etc
- Any name is hashed as a key with SHA-1
- The key is normalized as *r* between [0,1]
- The normalized key *r* is mapped to a virtual point on the unit circle
  - x = cos (2π*r*)
  - y = sin (2π*r*)

Storing and solving names

- A pair (key,value) is stored in the peer of depth *p* closest to its virtual point
- If this peer does not exist, the pair is stored in its first existing ascendant peer
- Storage is done in nodes being at tree depth ≤ *p*
- *P* is arbitrarily fixed at the overlay creation

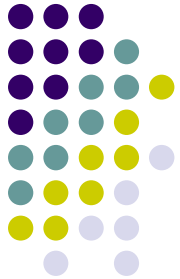


Steering



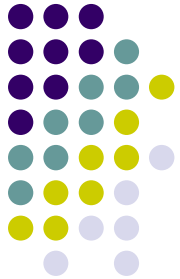
- If an @ becomes invalid, reply packets may contain new @ enabling intermediate peers to reroute on the fly until source peer sets new @
- Intermediate peers can use the DHT to get the new @
- This mecanism also enables efficient multicast capability

CLOAK operations



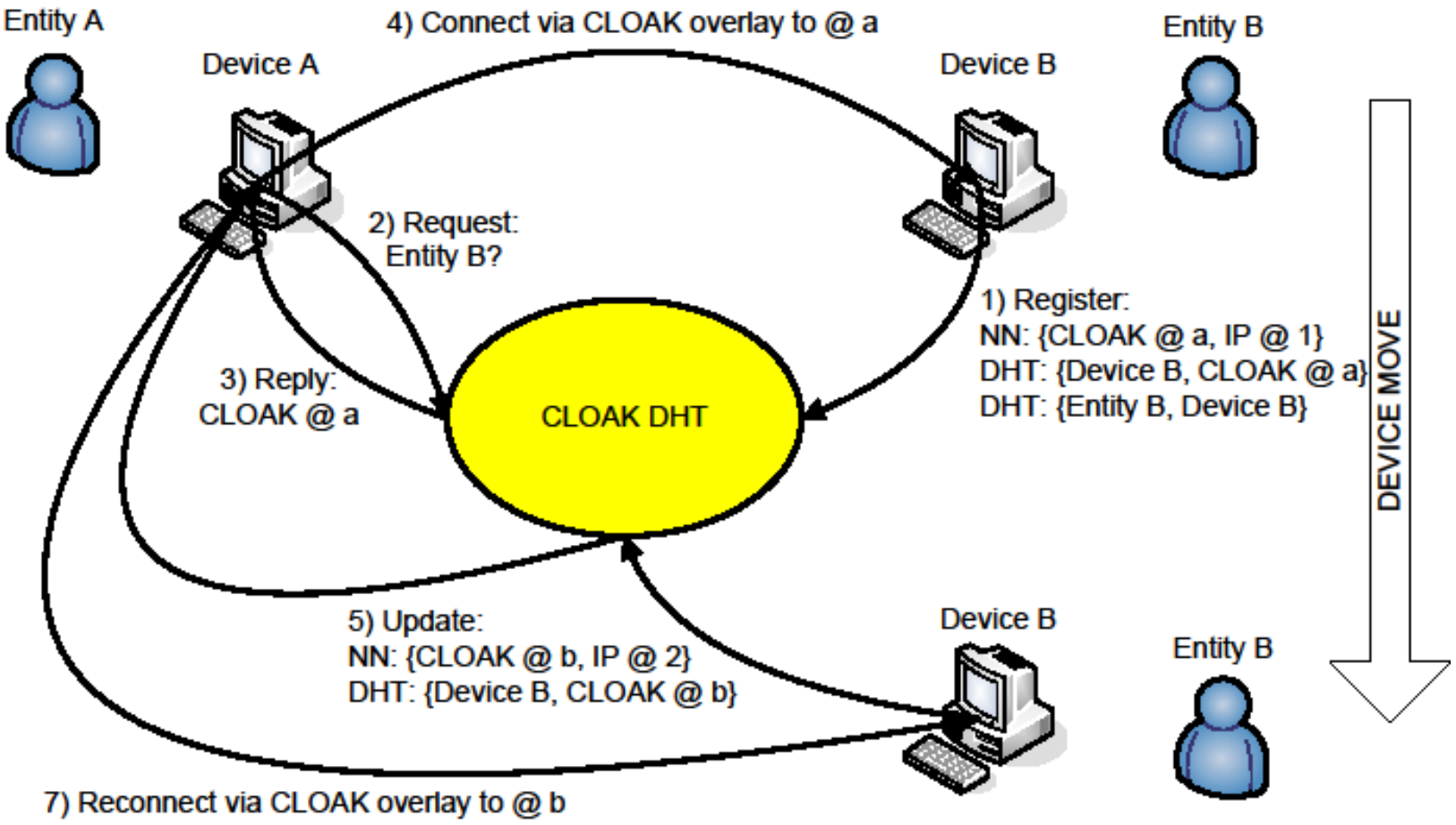
- Manager
  - *Bootstrap* into the overlay by setting transport layer connections to peers (neighbor peers)
  - Obtain an overlay address from a neighbor peer
  - *Identify* oneself in the overlay with unique device and entity names
- Application
  - Create a *session*
  - Start or wait for overlay layer connection(s)
  - Contact or accept 1(+) peer(s) to communicate with a given *interaction*
  - Send or receive data stream(s) to or from the session member(s)

Bootstrap

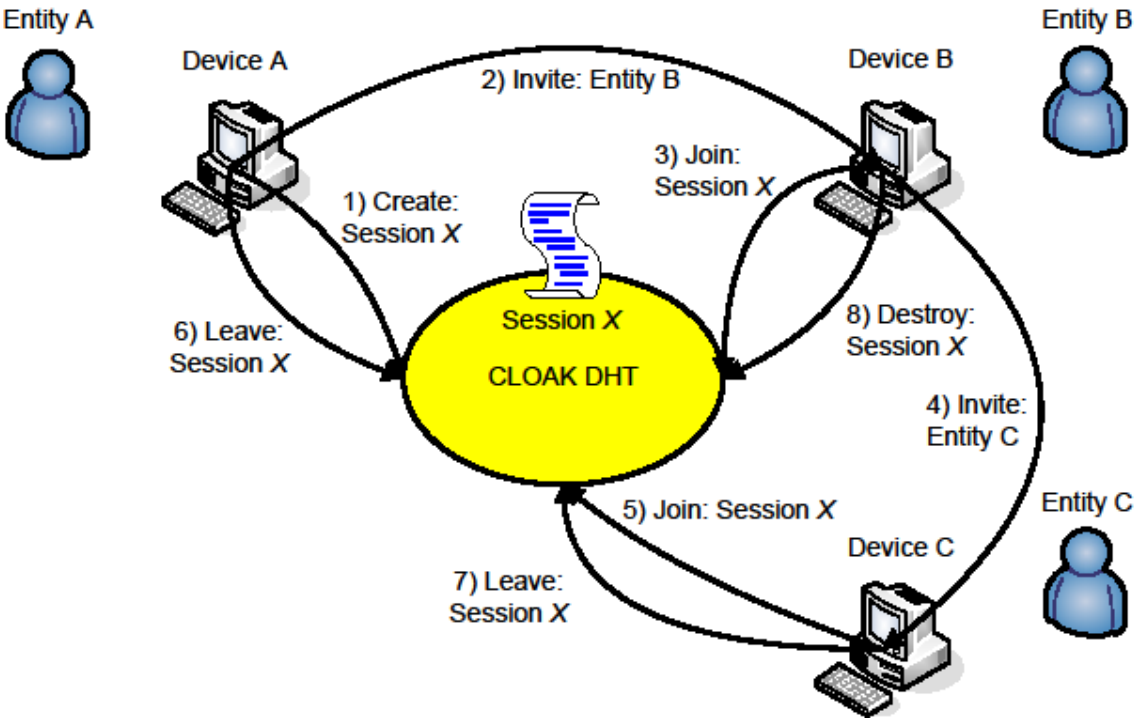


- A peer wishing to join an overlay must know
  - The overlay name or ID
  - The IP@, transport protocol n° and port n° of a peer already inside the overlay (called a *gate*)
- The overlay ID and gate information must be obtained by out-of-band means
  - Could be defined as an URL/URI
  - Carried on web pages, e-mails, text messages
- Overlays are isolated by design

Identification and localization



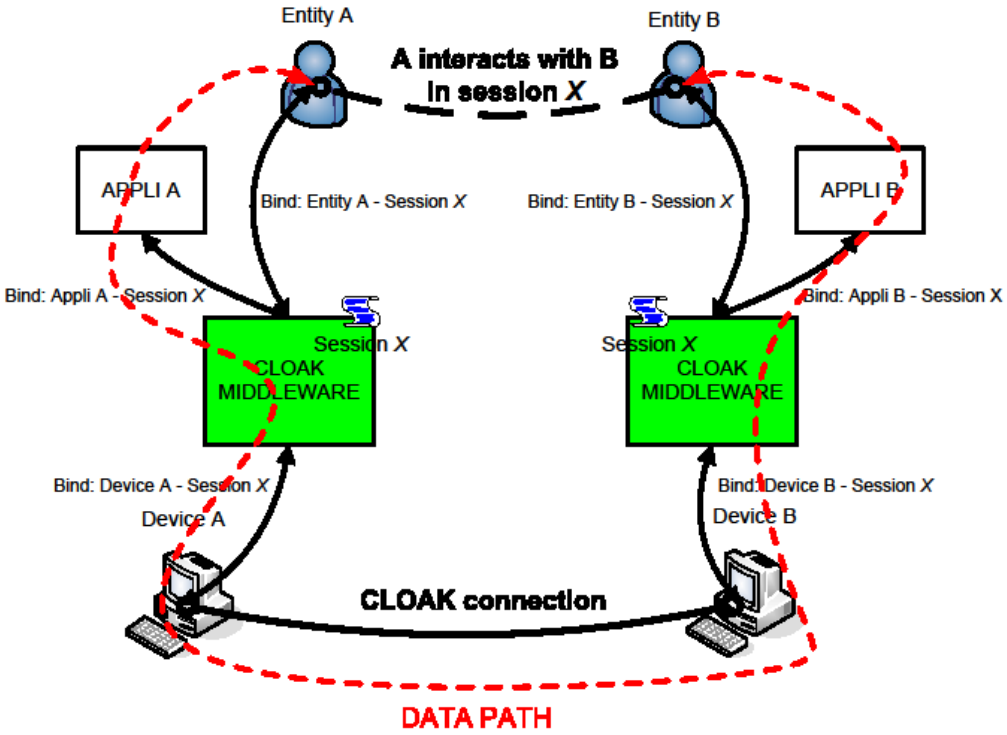
Session



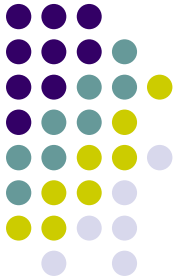
- A session manages a communication between 2+ overlay peers and can be stored in the DHT (esp. if both sides move)
- Ends when all peers have quit

Interaction

- An interaction defines an application layer protocol
- An overlay connection uses a triplet (entity, stream, application) at each endpoint
- All connections are stored in sessions

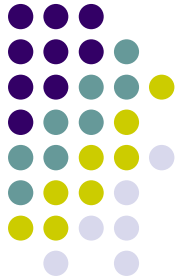


CLOAK architecture



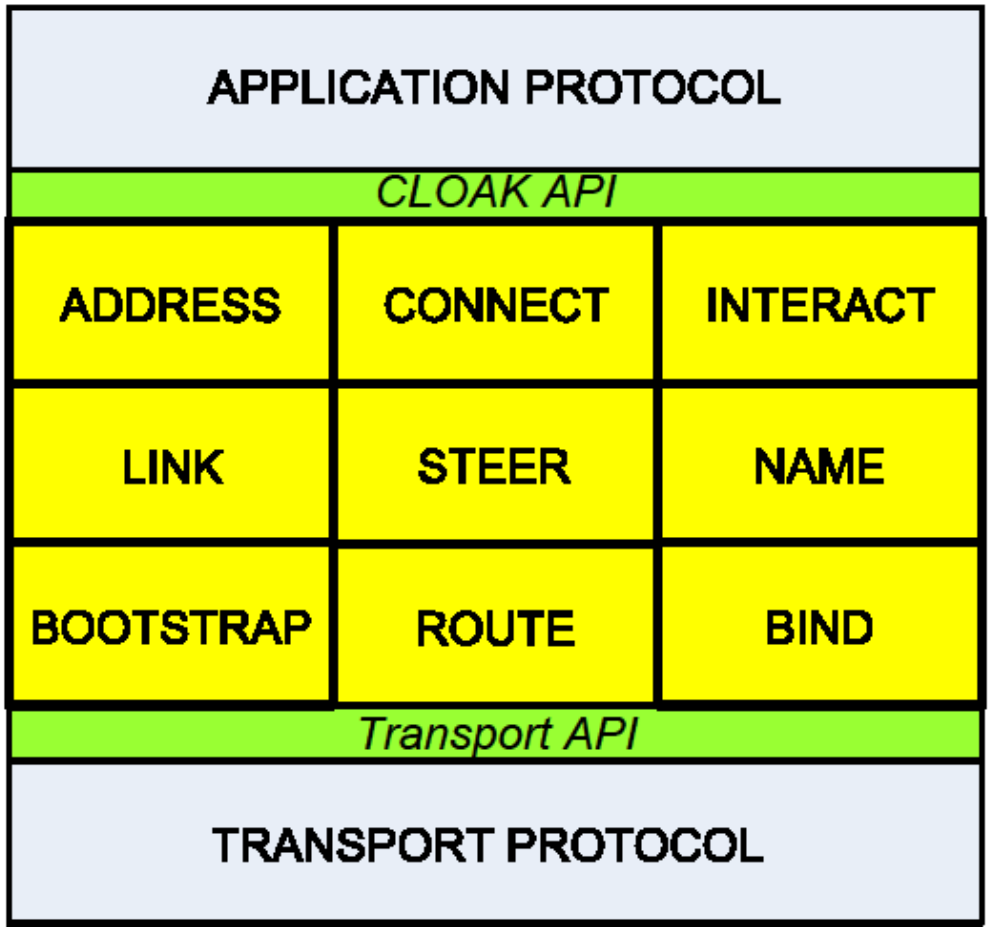
- Components
- Service blocks of the middleware
- Layers of connections
- Protocol stack
- Protocol headers and encapsulation
- Requirements

Components

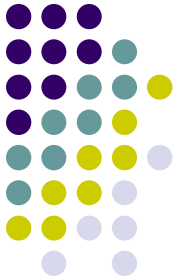


- A specific program called a *manager*
  - User interface for managing all the sessions of all overlays the entity / device belongs to
  - Service for maintaining the transport connections and routing the data
- A library to be used by applications
  - Implementing the CLOAK API calls

Block diagram

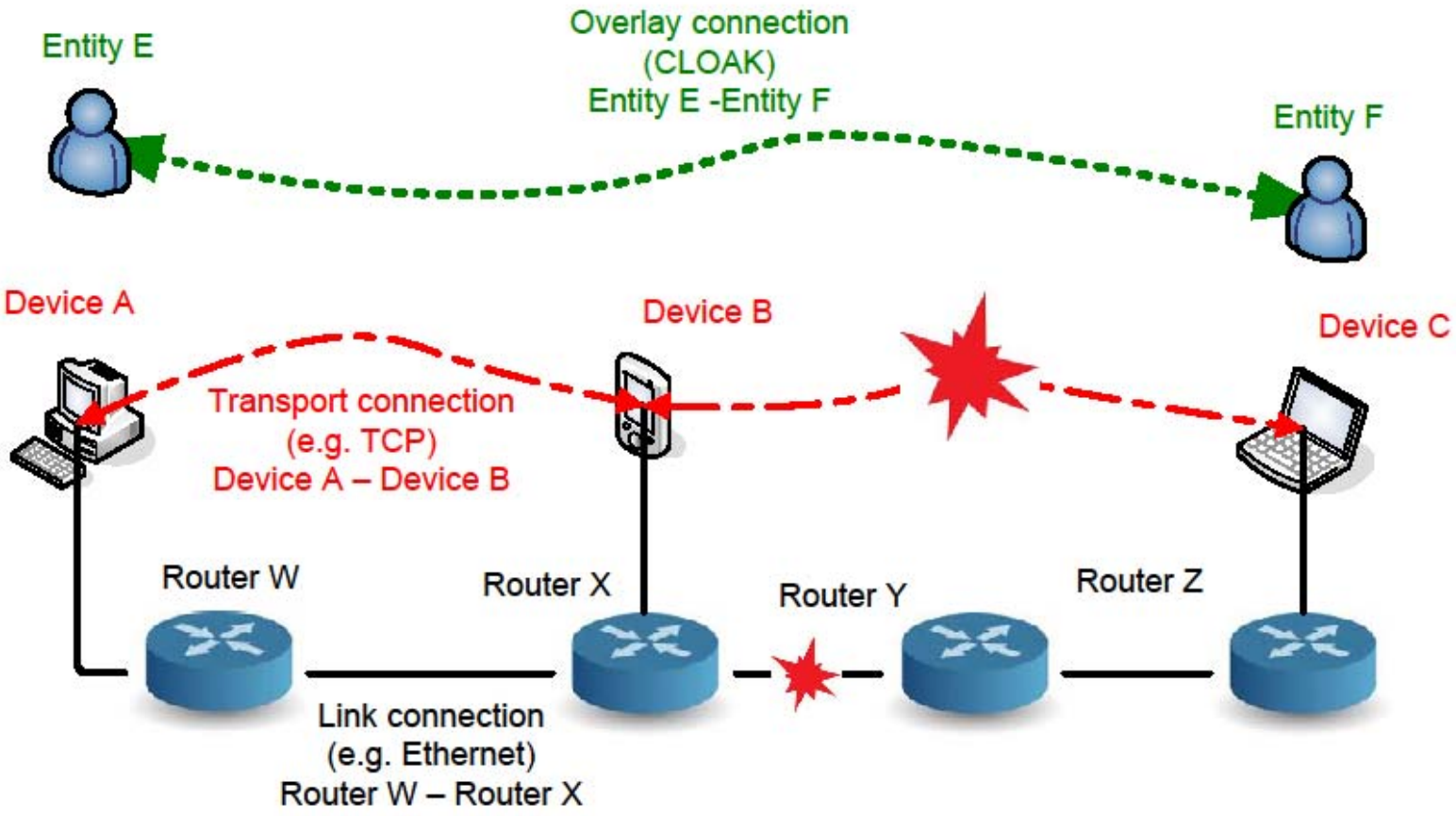


Services



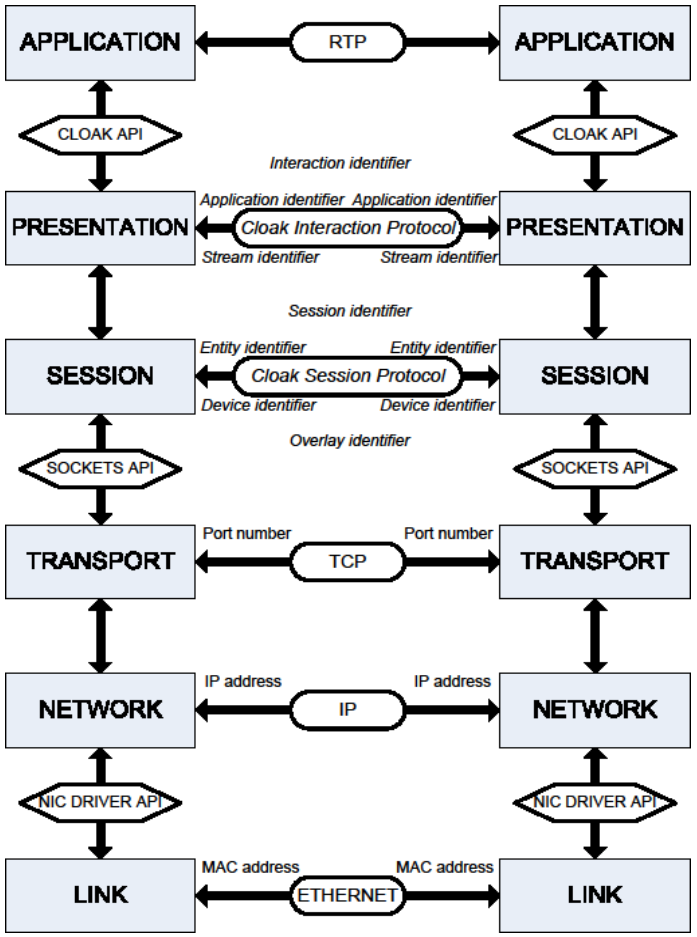
- Bootstrap
  - create a new or join an existing overlay
- Link
  - manage overlay links (i.e., transport layer connections) with neighbor peers
- Address
  - obtain an overlay @ from an addressing tree parent and distribute overlay @s to the addressing tree children
- Route
  - greedily route the overlay packets with the hyperbolic distance metric
- Steer
  - reroute overlay packets by using their device or entity identifiers to update their overlay destination @
- Connect
  - establish and manage overlay connections (i.e., CLOAK layer connections) to other entities
- Bind
  - query the DHT of the overlay
- Name
  - manage the identifiers used by the peer
- Interact
  - manage the bindings between the data streams and the applications

Connections

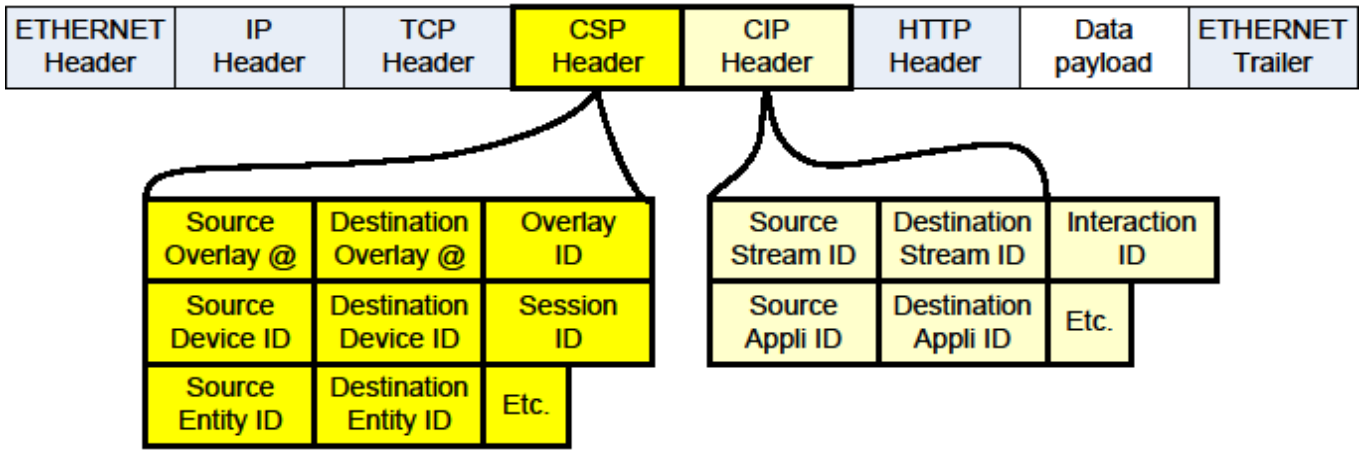


Stack

- Adds new protocols between transport and appli layer protocols
- Adds new identifiers and names
- Adds a new API
- Application is bound by virtual IDs but network IDs (IP@, protocol n°, port n°) can change

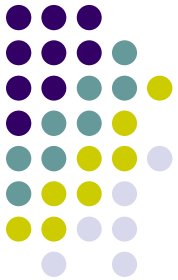


Encapsulation



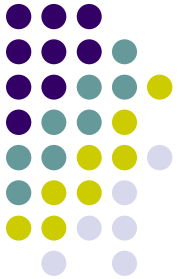
- 2 additional headers and protocols
- Overlay @ for routing in the overlay and moving devices
- Device ID for switching devices and moving entities
- Entity ID for switching entities
- Stream ID for virtual port numbering on the entity
- Application ID for selecting or switching applications

Requirements



- Applications must be modified
  - Because the comm. semantics are different
  - Unmodified applications shift work on the manager and make them awkward to use
- No modifications needed in networking equipments
- No modifications needed in host OS kernels
- No resources used in any host not belonging to the overlay

CLOAK usages



- Movable and transferable communications
- User level security
- Dynamic VPNs
- Anonymity layer
- Convergence layer for IPv4, NATs and IPv6
- Adaptive transport protocol switching
- Sandboxed use of new namespaces

Cloakable applications

- Messaging
  - Reachable anywhere
  - E2E privacy and authentication

Jabber

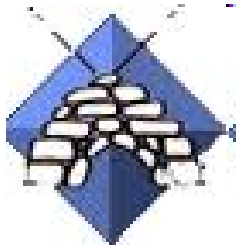
- Conferencing
  - A/V talk on the move
  - Multicast capability

GnuGk GNU Gatekeeper

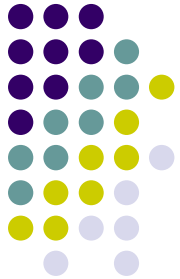
Cloakable applications cont'd

- Sharing
  - Anonymity
  - E2E privacy

- Streaming
  - Listen / Watch on the move
  - Redirect stream on the fly to another device or user



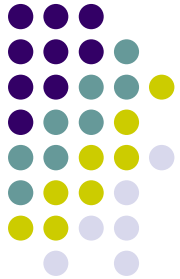
Comparison



- CLOAK is not a typical P2P sharing system
  - Data transits through intermediate peers
- Not a Grid computing system
  - No distributed computing services are provided
- Not a Cloud computing system
  - No user data storage services are provided

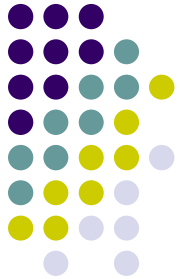
➔ *But P2Ps, Grids and Clouds can use CLOAK*

CLOAK evaluation



- Modeling
  - Deterministic
  - Random rolls for starting condition
- Simulation
  - Discrete event simulator
  - Peers have inter-arrival rate and session length
- Experiments
  - TCP chaining with debian systems

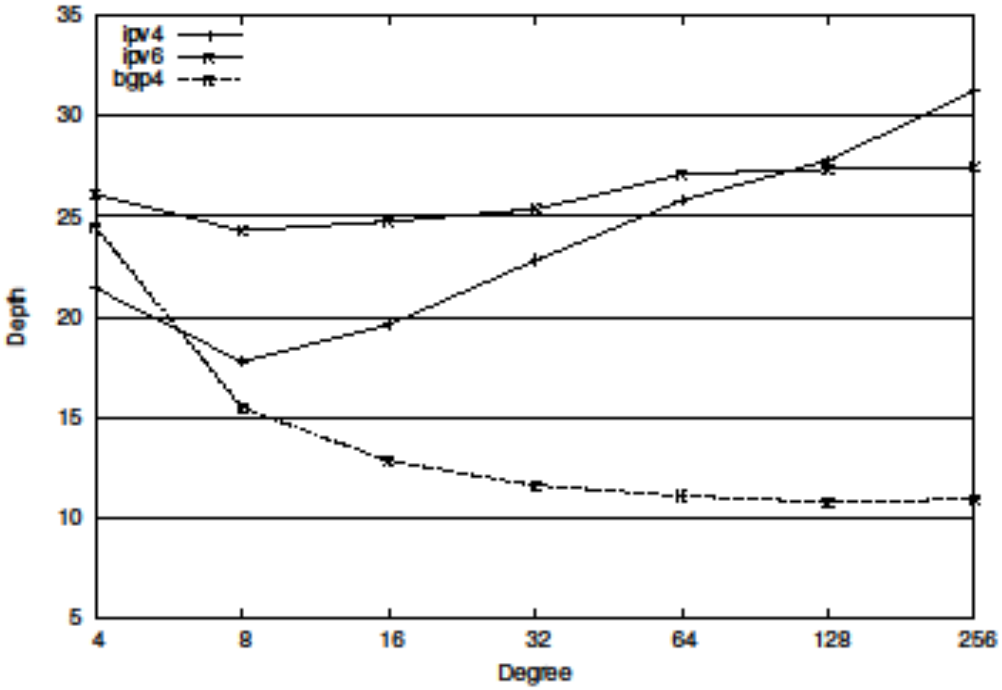
Modeling



- Static simulations in which each peer is always on
- Run on Internet maps (IPv4, IPv6 and BGP) where each node is a peer
- Degree of the addressing tree set from 4 to 256
- Addresses distributed by BFS from a randomly chosen peer
- One packet sent from every peer to every other peer (all paths are evaluated)

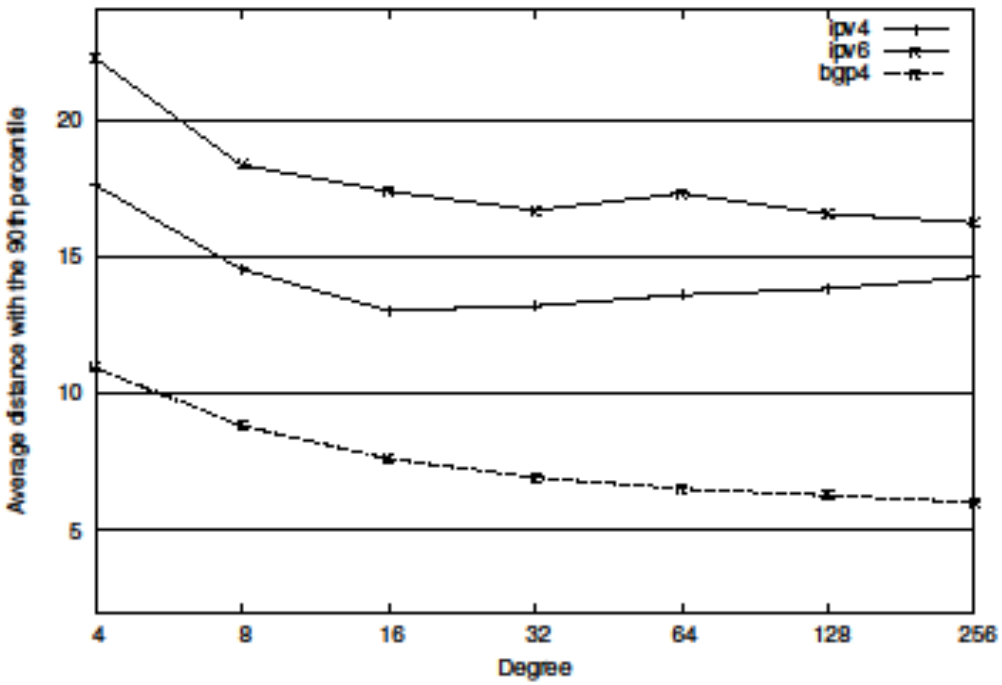
Depth

- Depth of the addressing tree for covering the maps
- Strongly depends on the value of the degree
- Strongly depends on the map type



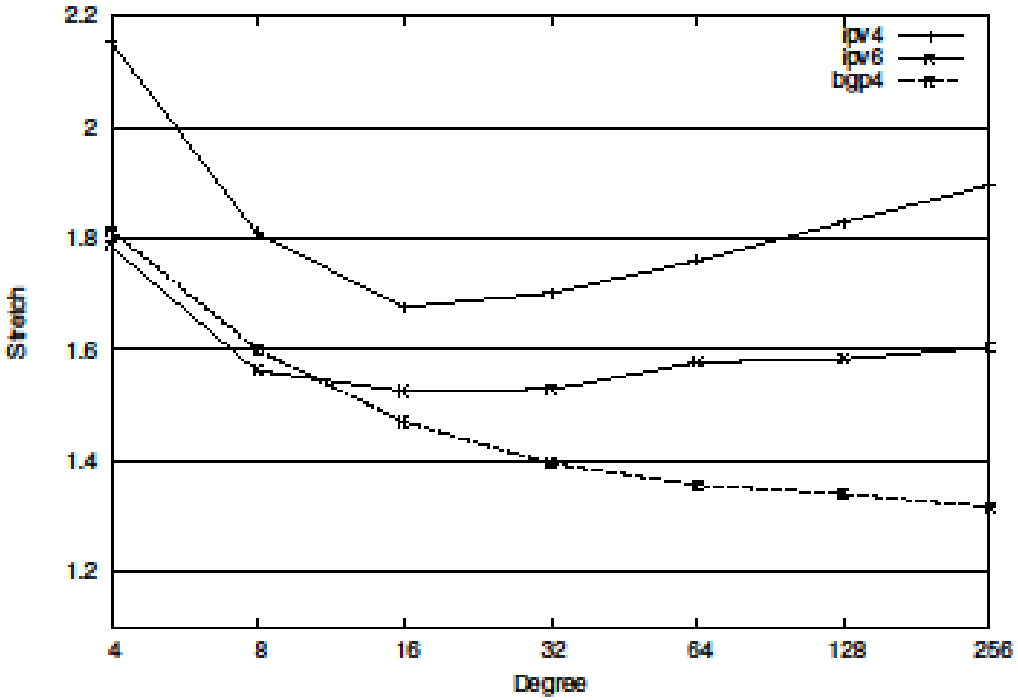
Path length

- Average path length between all nodes
- Proportional to the size of the map
- Correlated to the degree
- Diminishing return for IP maps with optimal degree value around 16



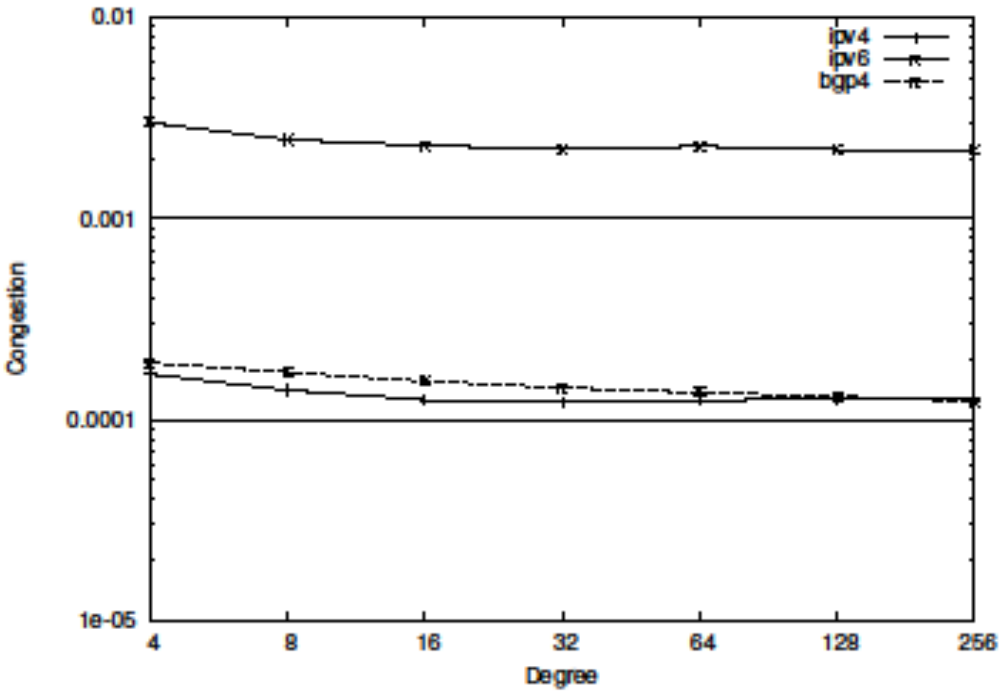
Stretch

- Remains below 2.2
- Correlated to the degree
- Strong diminishing return for IP maps with optimal degree value around 16

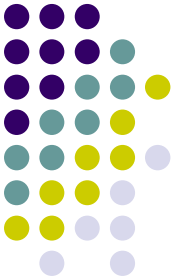


Congestion

- Average of the number of paths going through each peer
- Very low on average
- Proportional to the size of the map
- Uncorrelated to the degree



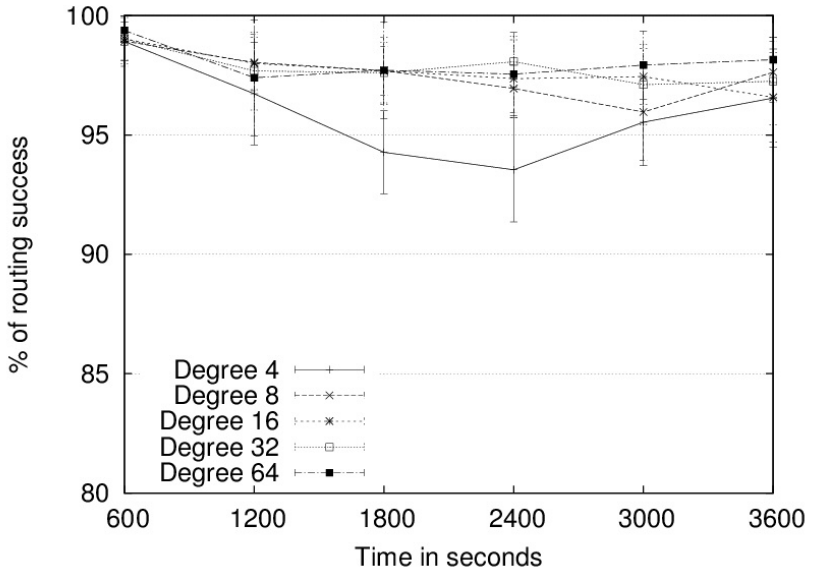
Stochastic DE simulations



- Peers have inter-arrival time and session length based on exponential distributions
- 30 new peers per minute, median peer session of 5 mn
- 2 stores/mn, 12 solves/mn
- 1h run duration, average of 20 runs
- No traffic model used because no throughput evaluation done yet

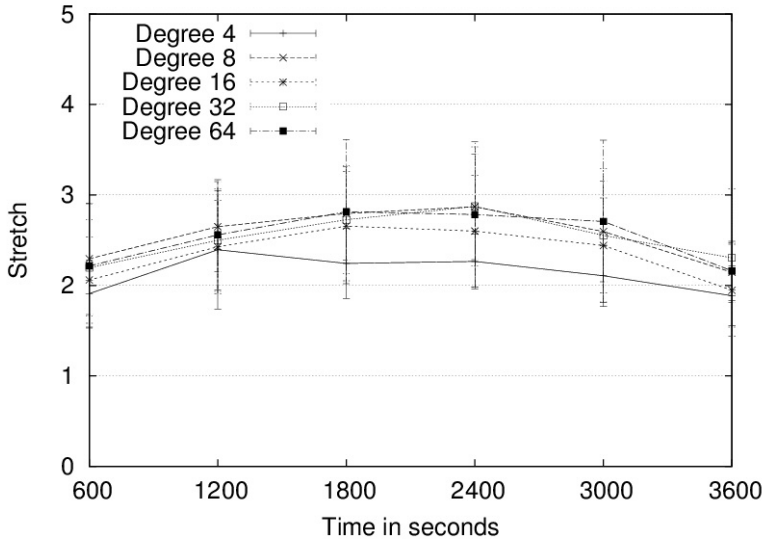
Routing success rate

- Remains high during the run despite high churn
- Worse for degree 4



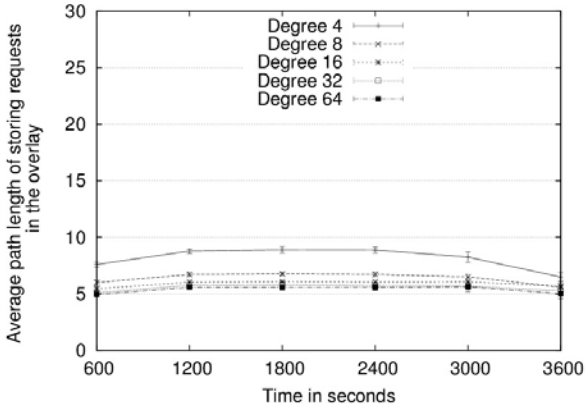
Stretch

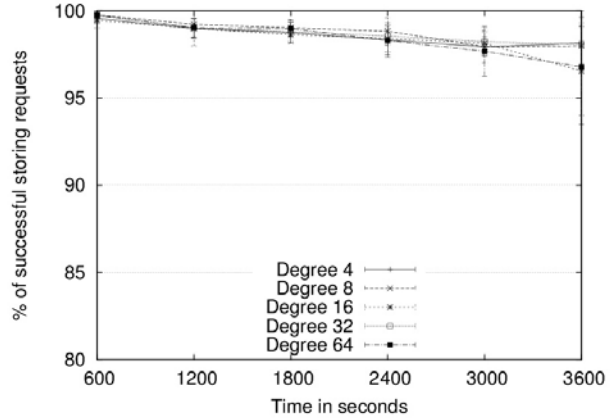
- Remains below 3
- Confirms results found in previous modeling deterministic simulations

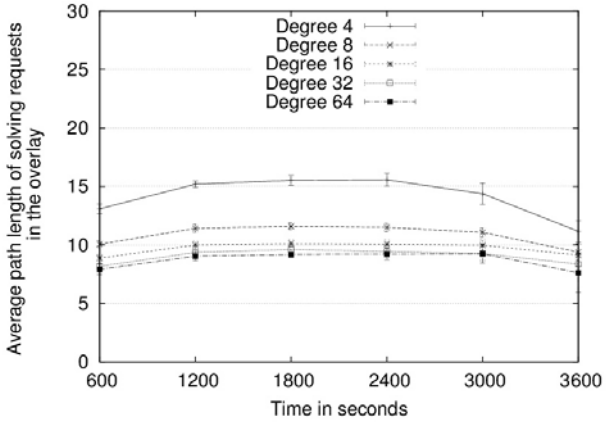


DHT

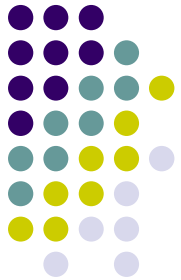
- Success rate is high
- Number of hops around five for storing a key







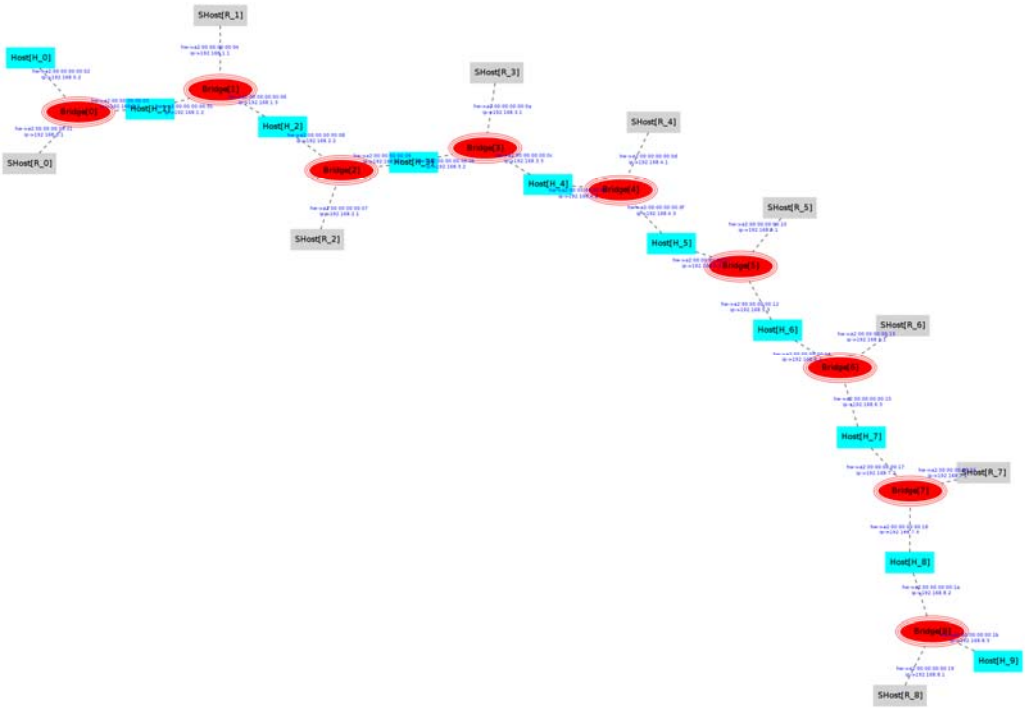
Experiments



- Evaluation of TCP chaining on data bitrate and application level delay
- How many TCP connections can be daisy chained?
- Use of a virtual network test bed created with virtual machines running on QEMU
- Packet size 1024B, message size 100MB

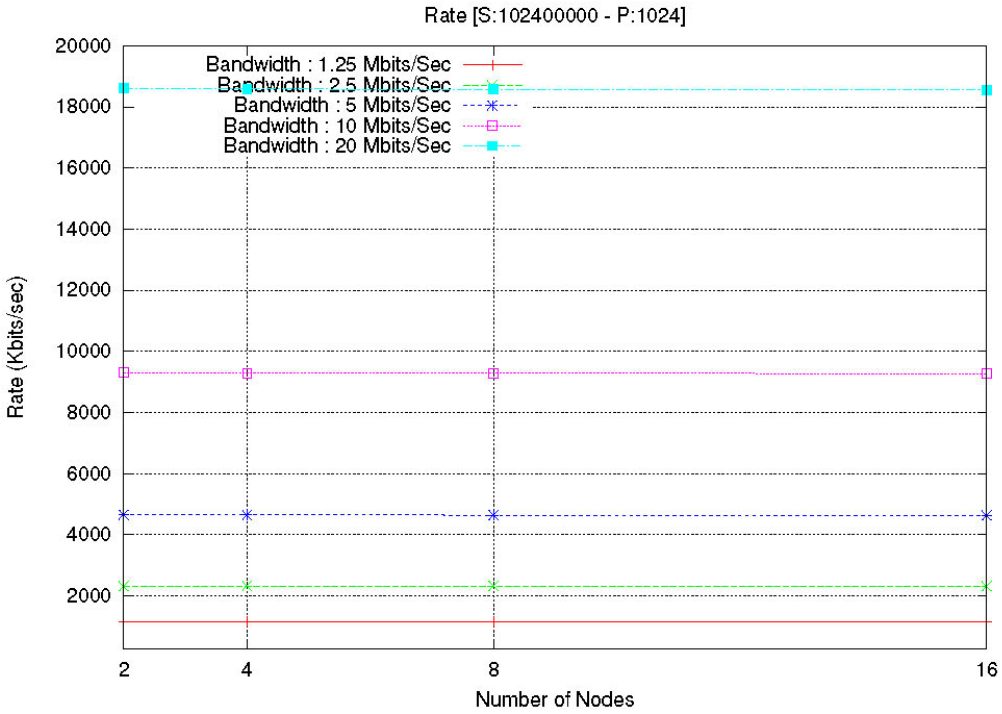
Network topology

- Chain topology
- From 2 to 16 hosts
- Use of QEMU VLANs to interconnect the hosts
- Automatic topology generation with our custom tool QNS



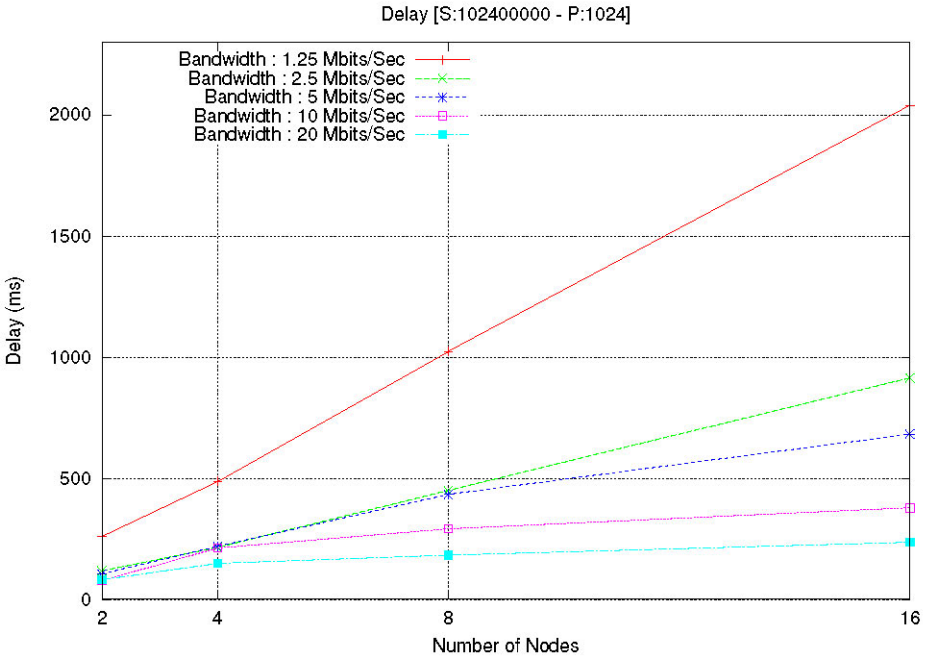
Bitrate

- Counts data rate only
- Remains the same up to 16 hosts
- Values just below the available BW
- No buffering at the application layer
- Average rates shown (reflects steady state)

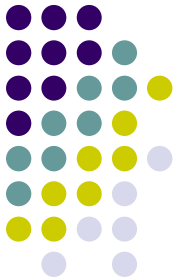


Delay

- Includes transmitting and processing delays
- Increases linearly when the BW is small
- Increases sub-linearly when the BW is high
- Can be a problem for real-time applications

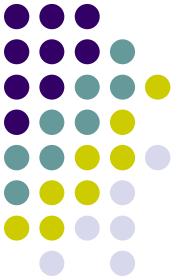


Conclusion



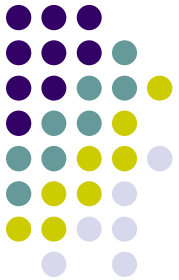
- Design of an overlay architecture that provides new features to applications
  - Mobility, flexibility, anonymity, security, autonomy
- Middleware that has nice properties
  - Scalable, reliable, usable
- Simulations on Internet maps show that overlay routing works ok
- Experiments show that transport layer connection chaining works ok

Future work



- Provide user space library and manager
- Port in test application (chat)
- Evaluate in a virtualized environment
- Provide kernel space integration
- Port in real applications
- Deploy in the Internet

Related work



- Host Identity Protocol (http://tools.ietf.org/html/rfc4423)
- SHIM6 (http://tools.ietf.org/html/rfc5533)
- Locator/Identifier Separation Protocol (http://tools.ietf.org/wg/lisp/)
- Transport next-generation (http://bford.info/tng)
- NUTSS (http://nutss.gforge.cis.cornell.edu/)

References

- **Overlay Addressing and Routing System Based on Hyperbolic Geometry.** Cyril Cassagnes, Telesphore Tiendrebeogo, David Bromberg, Damien Magoni, *ISCC'11 - IEEE Symposium on Computers and Communications*, to appear, June 28-July 1, 2011, Corfu, Greece.

- **Virtual Internet Connections Over Dynamic Peer-to-Peer Overlay Networks.** Telesphore Tiendrebeogo, Damien Magoni, Oumarou Sié. *INTERNET'11 – International Conference on Evolving Internet,* to appear, June 19-24, 2011, Luxembourg.

- **An Overlay Architecture for Achieving Total Flexibility in Internet Communications**. Cyril Cassagnes, David Bromberg, Damien Magoni. *AITM'10 - International Conference on Advanced Information Technologies for Management*, pp. 39-60, November 22-23, 2010, Wroclaw, Poland.

Questions?

magoni@ieee.org