Dynamic Peer-To-Peer Overlays for Voice Systems

Krishna Kishore Dhara
Venkatesh Krishnaswamy
Avaya Labs Research

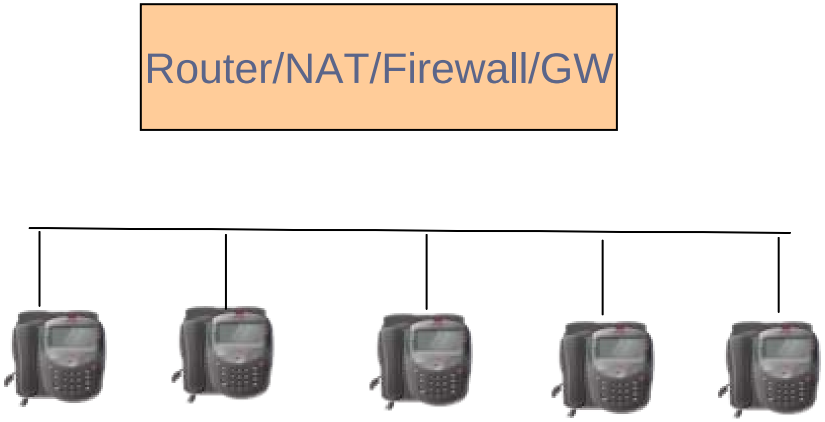
Salman Baset
Columbia University

IP Telephony | Contact Centers | Mobility | Services

Outline

- Background/Motivation
- Overlay Architecture
- P2P Overlays in SIP
- Examples of Overlays
- Summary/Conclusion

Value of P2P in the Enterprise



Branch/Small Office

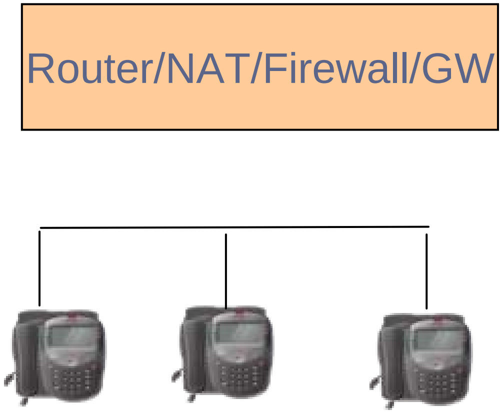
P2P Voice Solutions

- Based entirely on phones => low cost
- Plug and Play with minimal admin
- For IP-connected branches or small offices – no additional equipment required for these VOIP phones

Enterprise Services/Features are crucial

- Voice mail
- Conferencing
- Group features, Bridging, etc.

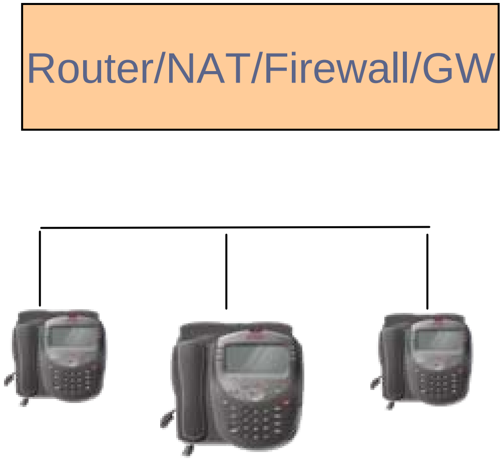
Implementing P2P Voice Systems



Flat

Broadcast/Multicast

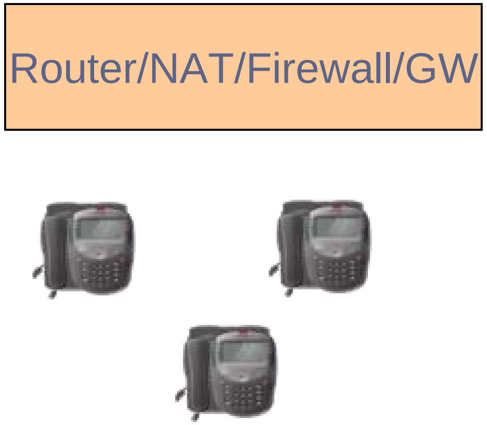
- Not scalable
- Small Office
- Simple



Hierarchical

Super Node/Proxy

- Somewhat scalable
- Not very simple

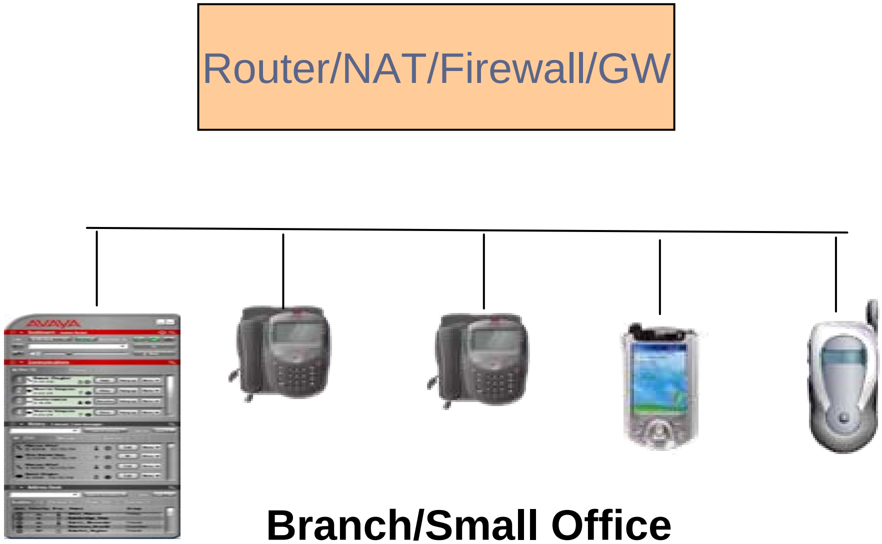


Structured

Distributed Hash Table

- Scalable
- Complex

Heterogeneous Enterprise Networks



Branch/Small Office

Heterogeneous devices have different

- network/bandwidth requirements
- processing
- security
- join/leave intervals

Users also have different

- preferences
- security – auth and trust mgmt
- multiple device identities

Services also have different

- network requirements
- data storage and processing
- security

P2P Voice Systems

Problem: How can we design a voice/communication system that can

- realize different capabilities and requirements of heterogeneous enterprise networks
- separate P2P properties from the underlying voice and transport protocol.

We propose

a layered framework that capture the device, user, and service overlays

a mechanism that decouples P2P overlay and the underlying voice protocol (SIP)

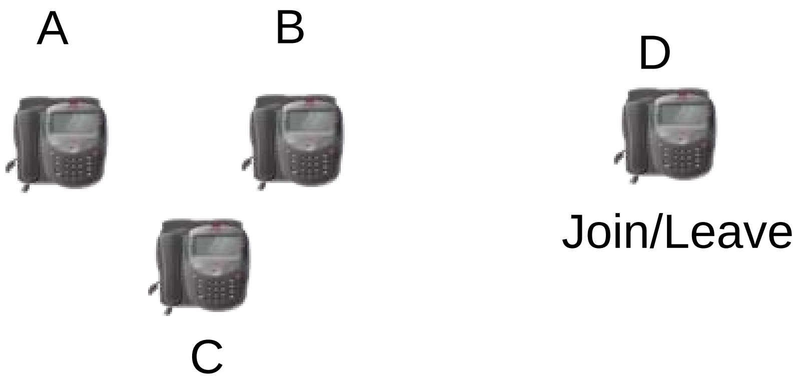
P2P Voice Systems – Current Approaches

- Skype, Avaya
  - Proprietary
- XMPP, JXTA – text based (XML) protocols
  - Need further exploration
- SIP P2P Systems (Kundan and Schulzrinne, Bryan et al)
  - Not modular overlays, close integration with SIP

A Layered Framework for P2P Systems

Physical Overlay:

Overlay peer connectivity, discovery, recovery



A

B

D

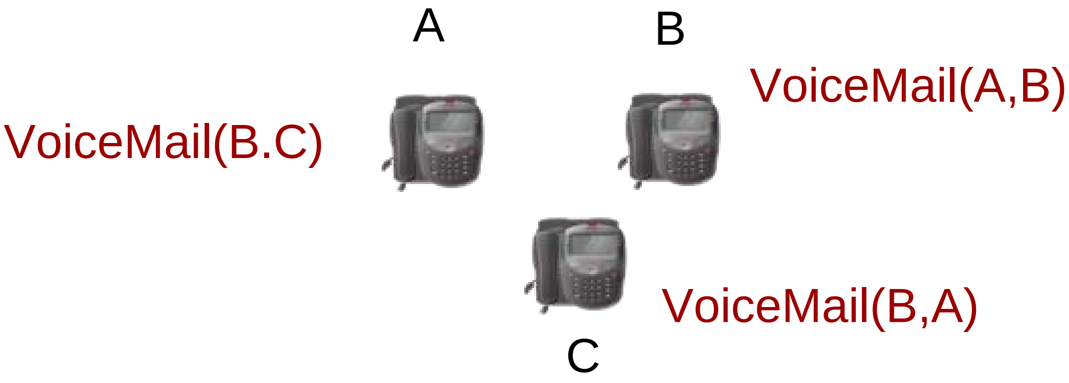
Join/Leave

C

Logical Overlay:

Implements device features, user features, and services

Constructed using physical overlay mechanisms.



A

B

VoiceMail(A,B)

VoiceMail(B.C)

VoiceMail(B,A)

C

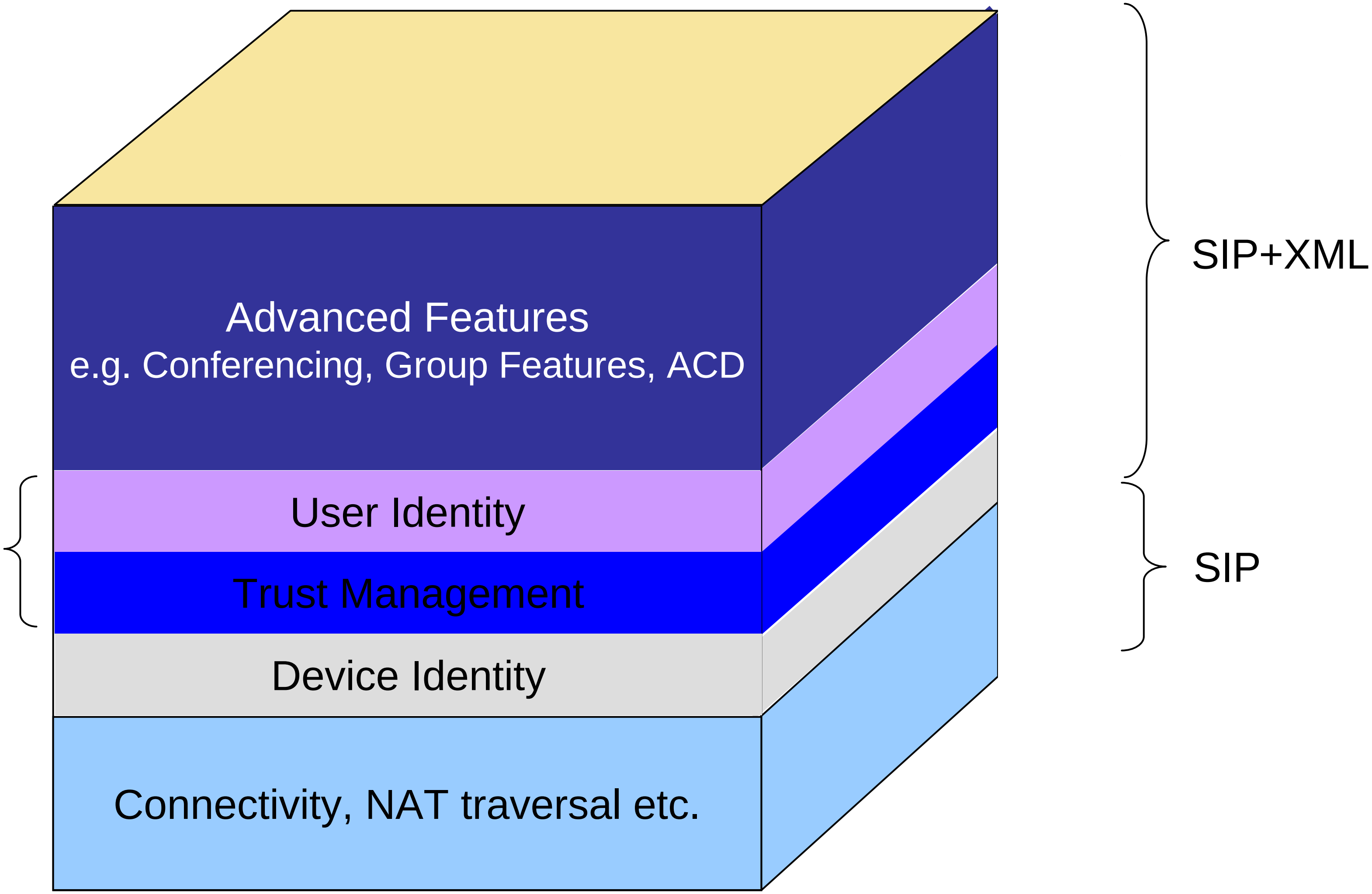
Distributing Voice Mail as a P2P Overlay.

P2P Over SIP

- Physical overlay uses SIP
  - Inherently P2P
  - Leverage mechanisms Routing, Authentication, etc.
  - Mature VOIP signaling model

- Logical overlay as XML bodies in SIP Messages
  - Prevents SIP protocol bloating
  - Separates P2P algorithm from protocol – therefore easier to craft an overlay structure that is optimized to the service being delivered

The Overlay Stack



Advanced Features
e.g. Conferencing, Group Features, ACD

User Identity

Trust Management

Device Identity

Connectivity, NAT traversal etc.

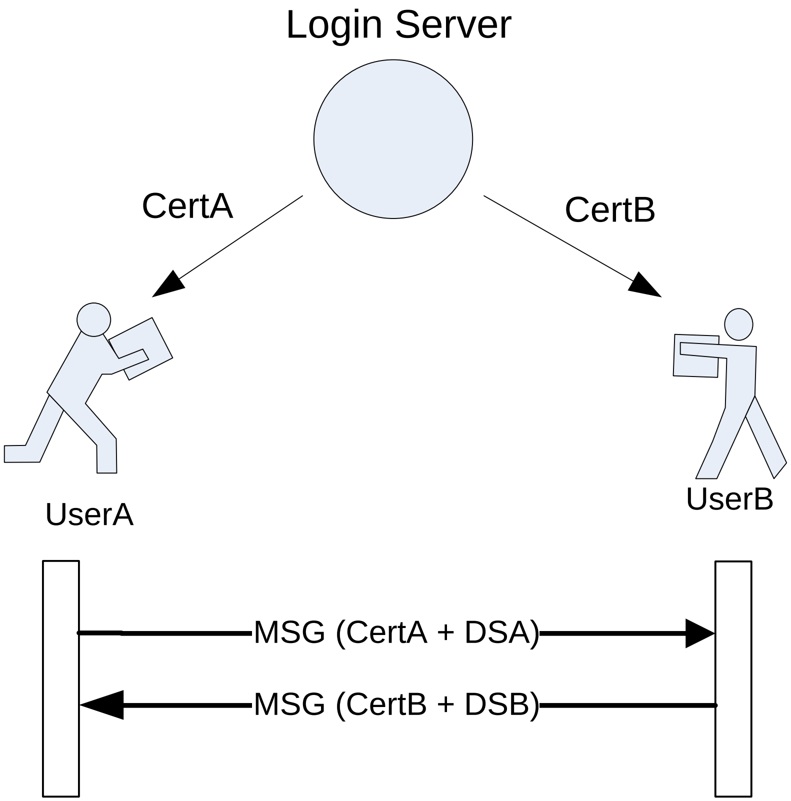
Examples from our prototype

SIP+XML

SIP

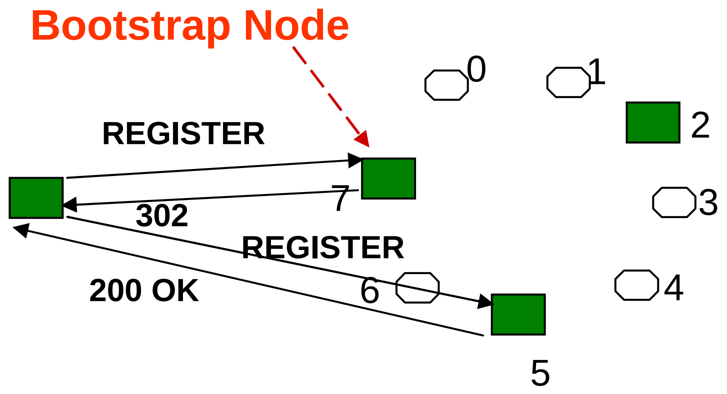
Example: Trust Management Overlay

- How can nodes trust each other?
- PKI-based solution
- Certify public key at login
  - User A: public key PuA, private key PrA
  - Login server: public key PuLS, private key PrLS
  - Certify user A public key (PuA) at login PrLS { PuA }
- Proof of Identity
  - Certified public key
  - Digital Signature



Example: User Overlay – Forming a (Chord) Structure

Each node is a: UA, Registrar, Proxy



```
REGISTER sip:atlanta.com SIP/2.0
From: sip:bob@atlanta;tag=11
Content-Type: appication/p2p+xml

<?xml version="1.0"?>
<P2Pxml>
<BootstrapRegRequest>
<NodeID>2</NodeID>
<NodeURL>sip:10.8.6.176</NodeURL>
<Certificate>Xj1...<trunacted></Certificate>
<Signature>v2R...<truncated></Signature>
</BootstrapRegRequest>
</P2Pxml>
```

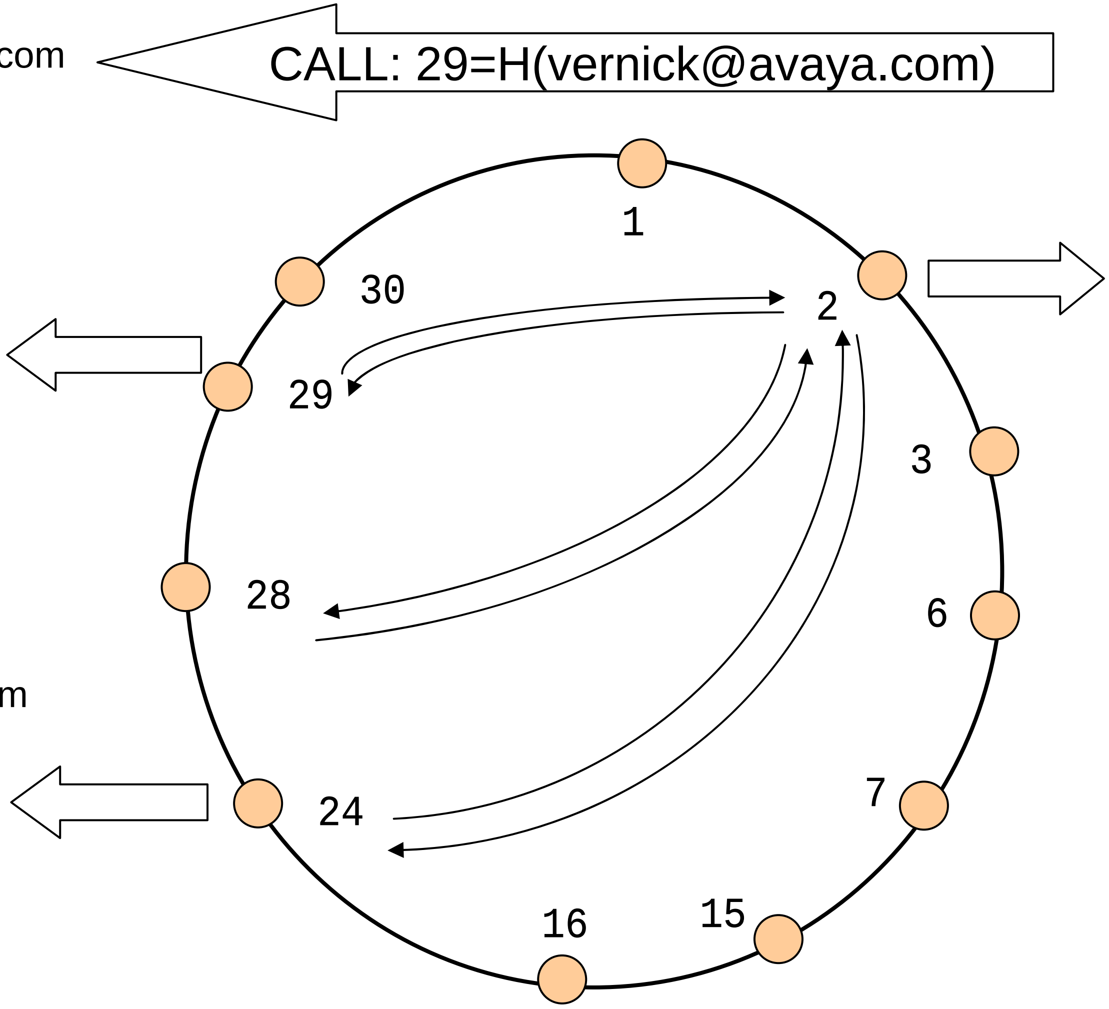
```
SIP/2.0 200 OK
From: sip:bob@atlanta.com;tag=11
Content-Type: application/p2p+xml

<?xml version="1.0"?>
<P2Pxml>
<BootstrapOK>
<NodeID>0</NodeID>
<Certificate>fFD...<truncated></Certificate>
<Signature>v2p...<truncated></Signature>
<SuccessorURL>sip:alice@atlanta.com:5060</SuccessorURL>
<SuccessorID>0</SuccessorID>
<PredecessorURL>sip:pred@atlanta.com:5060</PredecessorURL>
<PredecessorID>6</PredecessorID>
<RefreshRate>100</RefreshRate>
<SuccessorList>
</SuccessorList>
<FingerTable>
...<finger table info>
</FingerTable>
</BootstrapOK>
</P2Pxml>
```

Example: User Overlay: Locating Users

vernick@avaya.com

Key	node
29+1 = 30	30
29+2 =31	1
29+4 = 1	1
29+8 = 5	6
29+16=13	15


CALL: 29=H(vernick@avaya.com)

baset@avaya.com

Key	node
2+1 = 3	3
2+2 = 4	6
2+4 = 6	6
2+8 = 10	10
2+16=18	24

dhara@avaya.com

Key	node
24+1 = 25	28
24+2 = 26	28
24+4 = 28	28
24+8 = 32	1
24+16=8	15

Considerations in deploying P2P overlays

- Heterogeneity:
  - Heterogeneous nodes; may not be possible to map any “feature” to any node
  - Heterogeneous users; may not be possible to completely “flatten” user address space
  - User groups
  - Feature interactions
- Run-Time Overhead
  - Creating and maintaining overlay structures
  - Iterative/layered lookup
- Management and Administration

Many Open Issues

- Users
  - Mobility: Structures for “permanent” nodes and nodes that are mobile
  - Services for nodes/users that are not present
- Security
  - Authentication and Trust Management
  - Authorization and Encryption
- Network and NAT Traversal issues
  - Optimizations for bandwidth and connectivity
  - STUN,TURN, ICE for P2P systems
- Routing
  - Optimizations for finger table size, hops
- Storage

⇒ Can we leverage knowledge of enterprise network topology and user behaviors?

- e.g. Organizational, administrative or network domains
- e.g. Calling patterns, social networks

Conclusions

Summary: We presented

- a layered architecture for P2P voice systems
- a SIP P2P mechanism that separates the P2P overlays and the underlying signaling and media protocol
- two different overlay mechanism from our prototype implementation

Contributions: Our approach

- isolates concerns and restrictions at each layer
- allows choice of P2P protocol based on devices, users, and services with different properties
- allows dynamic swapping of P2P protocol