**Background and Motivations**
- Virtual Private Networks (VPNs) enforce privacy and authentication in IP communications
- Privacy desirable for end-to-end communication among users
- Virtual network desirable to connect multiple cloud instances
- Challenges in existing VPNs:
  - Configuration is difficult, error-prone
  - Node mobility and elasticity (e.g., phones, cloud VMs)
  - Performance overhead of centralized server gateways
- IPOP: user-defined, easy to configure, self-managed end-to-end VPNs connecting nodes through P2P links

**IPOP Core Features**
- User-level: works on a variety of operating systems and devices
  - Linux (Desktop/laptop, Android, OpenWRT), Windows
- Peer-to-peer links with integrated NAT traversal
- IPv4 and IPv6 virtual networking – supports existing applications
- User-friendly social network based peer discovery
- User-transparent certificate exchange and network configuration
- Build upon standards for messaging and NAT traversal
  - XMPP, STUN, TURN
- Modular and extensible

**Use Cases and Users**
- Virtual clusters, platforms across multiple clouds
  - ConPaaS, Kangaroo (PaaS)
- Software-defined networking testbed
  - PRAGMA-ENT
- Collaborative environments for e-Science / PRAGMA
  - GRAPLeR lake ecology modeling expedition (distributed HTCondor cluster)
  - Rocks (PRAGMA Boot)
- Disaster management (Navy/DoD)

**Software: Status and Future Work**
- Major software design re-factor: Current release: v16.01
  - Software reuse (Google libjingle), standards
  - Separation of concerns: datapath/control modules
  - Extensibility to support different overlay topologies
  - Support for mobile devices (Android), embedded (Raspberry Pi, Intel Edison), wireless routers (OpenWRT)
  - Multi-hop overlay routing, dynamic source routing
  - Switch mode (Layer 2, ARP) – multicast
  - Structured overlay GVPN controller
  - Google Summer of Code ‘15 Participant
- Ongoing/future work:
  - Support for DHCP, DNS
  - OpenFlow virtual switches for packet capture/injection
  - Bootstrapping: NAT traversal through social peers

**Architecture Overview**

**Switch Mode:**
- Virtual layer 2 switch – client in the same “flat” virtual network subnet
- Transparent to applications – no modifications necessary to client/server

**Major modules:**
- IPOP-TinCan: sets up TinCan links; tunnels, forwards packets
- Modular Controller: manages link start/trim; maps IP addresses; overlay routing
- IPOP-Tap: interfaces with O/S for packet capture/injection

**Controller/TinCan API:**
- Different VPNs for different use cases (SocialVPN, GroupVPN)

**Social Network Bindings and Peer Discovery:**
- XMPP Protocol – support for existing OSNs and open-source (ejabberd)

**NAT Traversal:**
- STUN (UDP hole-punching), TURN (relay through public node)

**SocialVPN and GroupVPN**
Two IPOP controllers maintained by the project

**SocialVPN:**
- Links social users to their friends
- Use cases: remote desktop, streaming, gaming, new social apps
- Individual user-centric: each user manages their own links
- Technical challenge – managing address allocation
- IPv4 addresses are dynamically assigned/translated

**GroupVPN:**
- Links multiple devices together in clusters
- Use cases: inter-cloud, ad-hoc virtual clusters
- Per-group: devices belonging to a group connected all-to-all
- Addresses are not translated