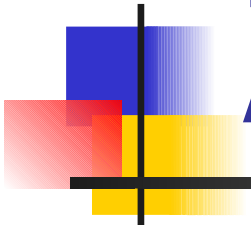
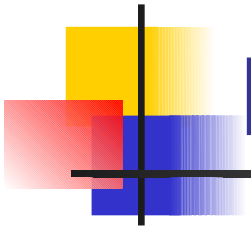


Popeye – Fine-grained Network Access Control for Mobile Users

Mike Chen, Barbara Hohlt, and Tal Lavian
{mikechen, hohltb, tlavian}@cs.berkeley.edu

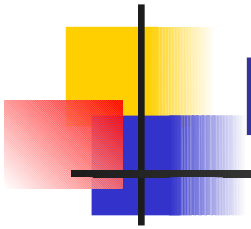
UC Berkeley
CS294-1 Mobile Computing and Wireless Networking

http://www.cs.berkeley.edu/~mikechen/research/popeye.html



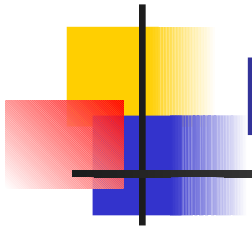
Motivation

- Existing Intranets don't support mobile user well while enforcing network access control
  - option 1: don't allow visitors to use the network
  - option 2: build a special visitor network that only has Internet access (no Intranet)
  - option 3: use SPINACH II, a Linux software router
    - only provides all-or-nothing access control
    - performance bottleneck when scaling to lots of users
- Existing Intranets don't support fine-grained protection from insiders
  - everyone has the same network-level access privilege and can reach all the machines on the Intranet
  - a growing problem as machines on the Intranet can be compromised through virus (e.g. 2000/10 Microsoft break in)



Project Goals

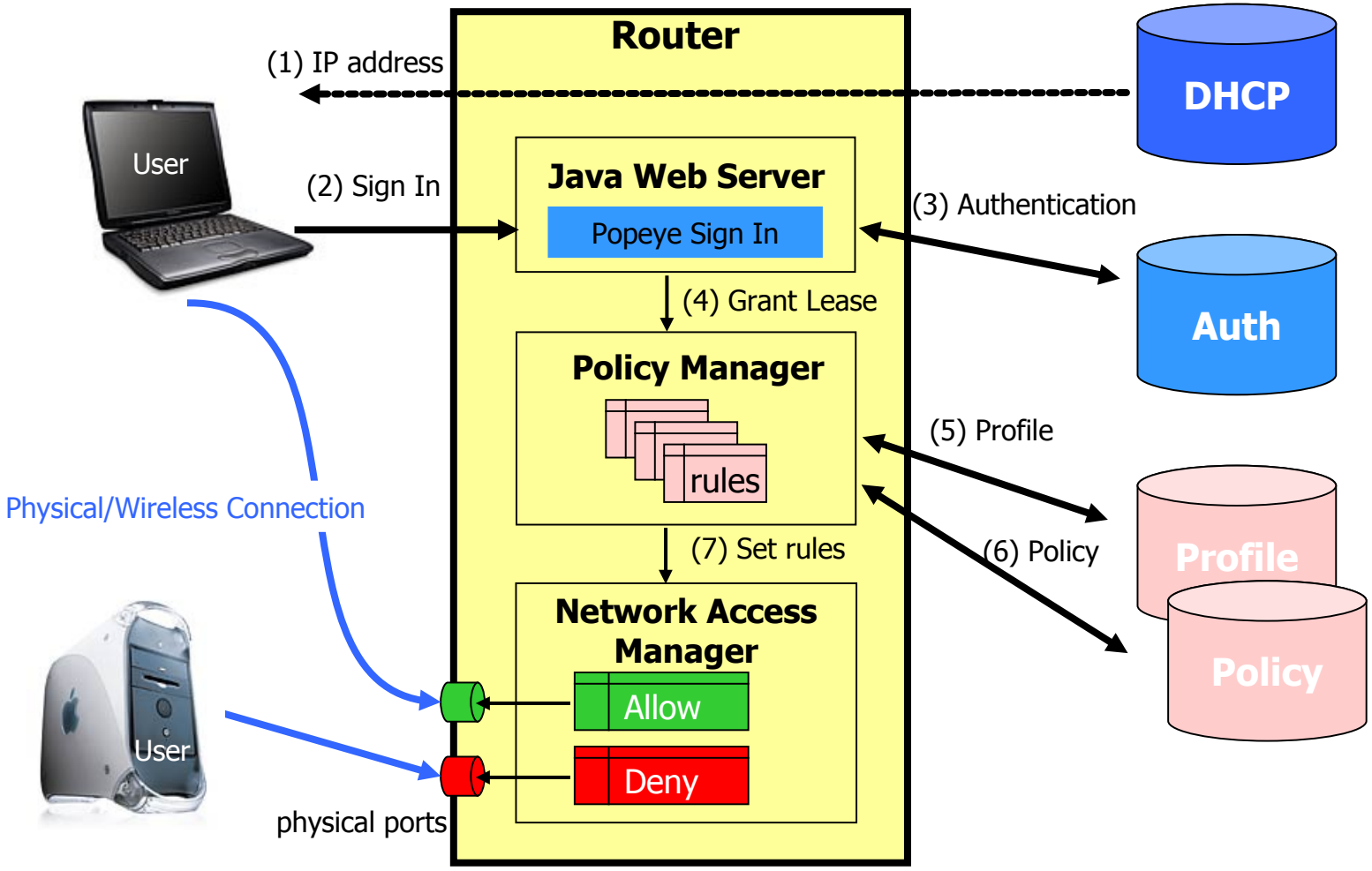
- Fine-grain access control and QoS
  - user-specific and app-specific
- Easy to manage
  - Policy Manager simplifies policy specification
  - system-wide, group-wide, and user-specific access control and QoS policy
- Modular design
  - support existing authentication servers (e.g. RADIUS, PKI, Kerberos) and user profile servers (e.g. LDAP)
  - enable different modules to be managed by different admins
- Scalable to lots of users
  - access control should impose minimal performance impact
- Easy to use

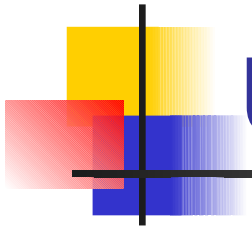


Design

- Authentication Manager and Profile Manager
  - stores the user credentials such as passwords and authenticates users
  - and stores the user profiles
- Policy Manager
  - stores policies and generates access control rules given a profile
    - system-wide (e.g. no Napster traffic)
    - group-wide (e.g. visitors only get Internet access)
    - user-specific (e.g. Adj gets the best QoS for video conferencing apps)
- Network Access Manager
  - configures the routers to enforce policies on *packets* and *physical ports*
  - e.g. setup packet filtering rules, move users to appropriate VLANs

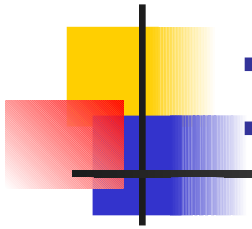
Architecture





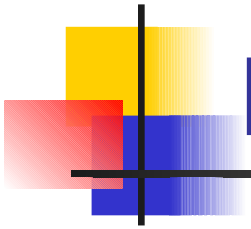
Usage Scenario

- Sign In
  - a mobile user gets an IP address through DHCP, and authenticates to the web server
- Authentication
  - the web server validates the user with the Authentication Server and grants a lease to the user
  - records the physical port and the MAC address of the user
- Authorization
  - the Policy Manager gets the user's profile and policies to generate access rules
- Access Control
  - the Network Access Manager enforces the access rules



Implementation

- Nortel Accelar programmable routers with Java support
  - supports up to 384 physical ports
    - to prevent IP/MAC spoofing, each user is required to be directly connected to a physical port
  - Java is used to configure the routers to setup the packet filters and QoS rules
  - actual packets are routed at *wire speed*
- IP packet filtering
  - protocol specific (e.g. HTTP only)
  - IP address specific (e.g. no access to the payroll subnet)
- VLAN access control
  - give each VLAN different access level (e.g. Intranet vs. Internet)
  - move users to the VLANs with the proper access



Discussion

- Performance
  - latency of VLAN setup: 2.0 seconds
  - latency/throughput is wire speed: switched 100Mbps
- Implemented VLAN-based access control
  - current Accelar routers doesn't support IP filtering
- Some support for wireless clients
  - MAC spoofing is a problem as WLAN is a shared LAN, and anyone can sniff the MAC address
- Usability
  - web interface is easy to use
  - hack: DHCP lease is set to really short so the client would quickly renew its lease once we move it to a new VLAN

Future Work

- Support wireless clients
  - to support per-user access control and prevent MAC spoofing, base stations must ensure that MAC address can not be forged (802.11 WEP is not adequate)
  - another option is to deploy VPN (e.g. IP-sec), but it has significant $$$, performance, and management overhead
- Design (or reuse) a policy language and build an UI for the Policy Manager
- Integrate packet filtering once the routers support it through the Java interface
- Support existing authentication and directory servers
- Deploy it on the new UCB campus-wide wireless network

The following are extra slides and diagrams to be used in our paper.

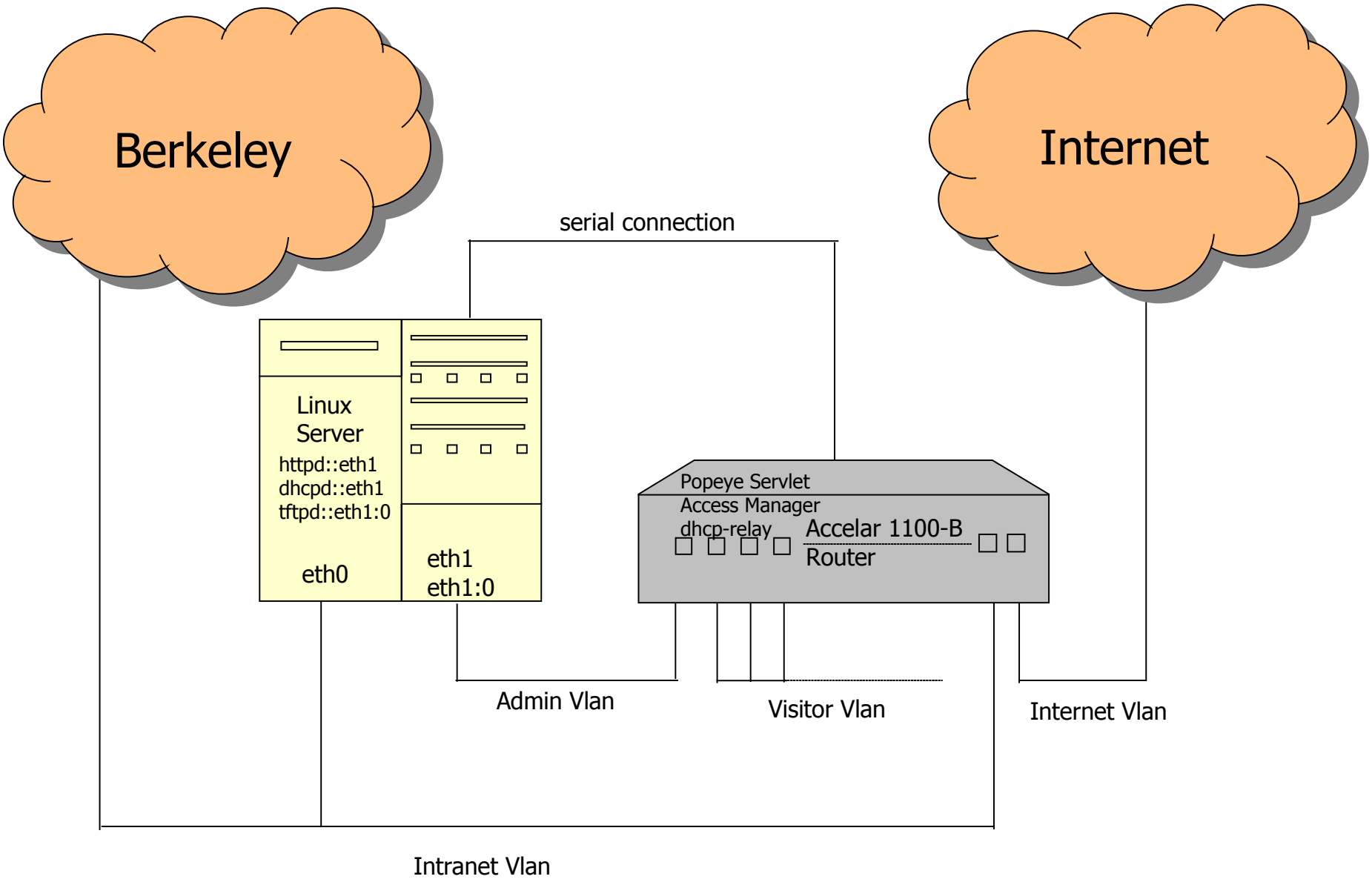
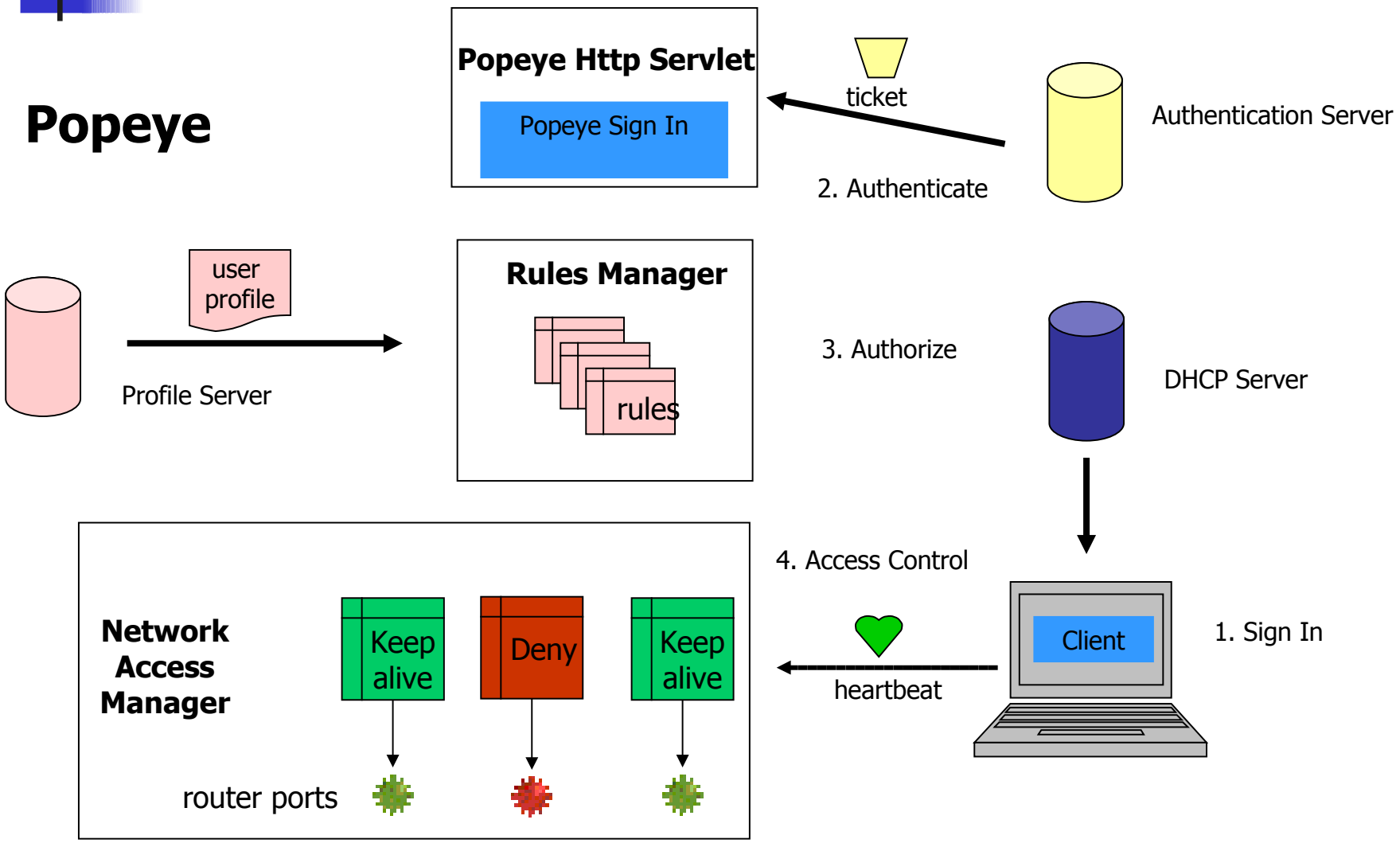
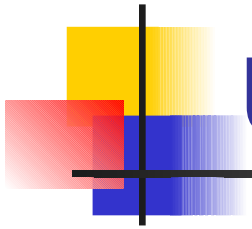

Berkeley
Internet
serial connection
Linux
Server
httpd::eth1
dhcpd::eth1
tftpd::eth1:0
eth0
eth1
eth1:0
Popeye Servlet
Access Manager
dhcp-relay
Accelar 1100-B
Router
Admin Vlan
Visitor Vlan
Internet Vlan
Intranet Vlan

Figure 2

System Architecture





Usage Scenario 2

- Sign In – a mobile client gets an ip address, goes to the Popeye web page, and enters name and password
- Authentication – the Popeye servlet validates the new visitor with the Authentication Server
- Authorization – the Popeye rules manager gets the visitor's profile from the Profile server and generates security rules for this visitor
- Access Control – the Popeye network access manager applies the security rules on the visitor's physical port, forks a keep alive thread which listens for heartbeats, and periodically evaluates the visitor's access

Port Table

Index
numVlanIds
vlanIds
Type
discardTaggedFrames
discardUntaggedFrames
defaultVlanId
performTagging

Net to Media Table

ifIndex
physAddress
netAddress
type

Interfaces Table

Index
Descr
Type
Mtu
Speed
physicalAddress
adminStatus
operStatus
lastChange
inOctets
inUcastPkts
outUcastPkts
outNUcastPkts
outDiscards
outErrors
outQLen
specific

Arp Table

ifIndex
doProxy
doResp

Vlan Table

Id
Name
Color
highPriority
routingEnable
ifIndex
Action
Result
stgId
Type
portMembers
activeMembers
staticMembers
notAllowToJoin
protocolId
subnetAddr
subnetMask
agingTime
macAddress
rowStatus
igmpSnoopEnable
igmpSnoopReportProxyEnable
igmpSnoopRobustness
igmpSnoopQueryInterval
igmpSnoopMRouter
userDefinedPid
igmpSnoopActiveMRouterPort
protocolIds
igmpSnoopActiveQuerier
igmpSnoopMRouterExpiration
igmpSnoopQuerierPort

Figure 3