



Enterprise QoS

Tim Chung

Google Corporate Netops Architecture

Nanog 49


June 15th, 2010





Agenda

- Challenges
- Solutions
- Operations
- Best Practices



Note:

This talk pertains to Google enterprise network only, not google.com

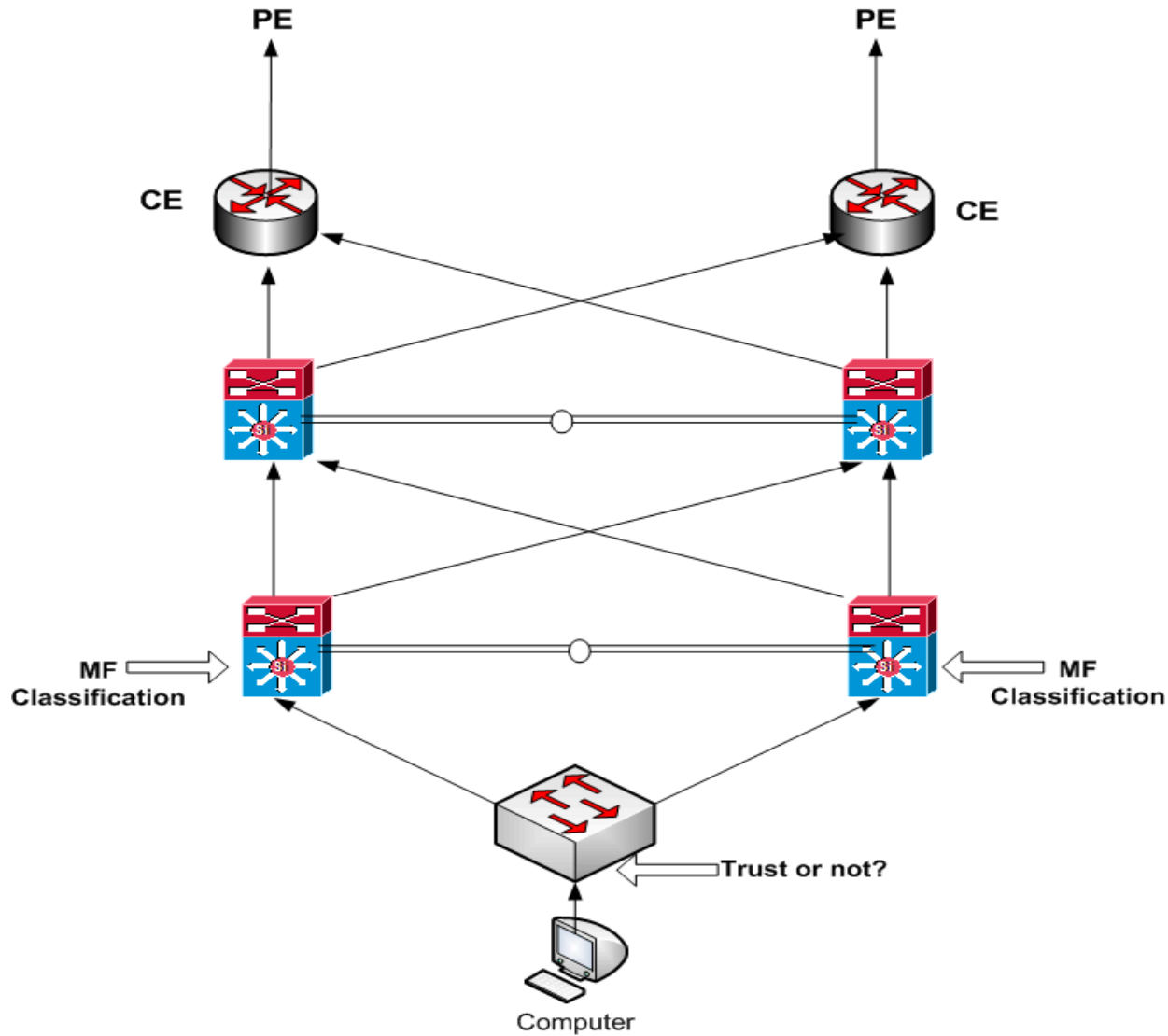


Challenges

- WAN connections are bandwidth bottlenecks
- QoS essential for latency and jitter sensitive real time apps
- Many applications require different QoS level
- Application classification granularity necessary on WAN Links
- Interaction with different MPLS provider requires different egress policies
- Multi-Vendor network
- Diverse Hardware: 4 Queue vs 8 Queue systems
- QoS over IPSec poses interesting challenge
- Performance: Need line rate Tput
- QoS MF Classification changes/update on weekly basis
- Statistical analysis for capacity planning



Classification as Close to Edge as Possible

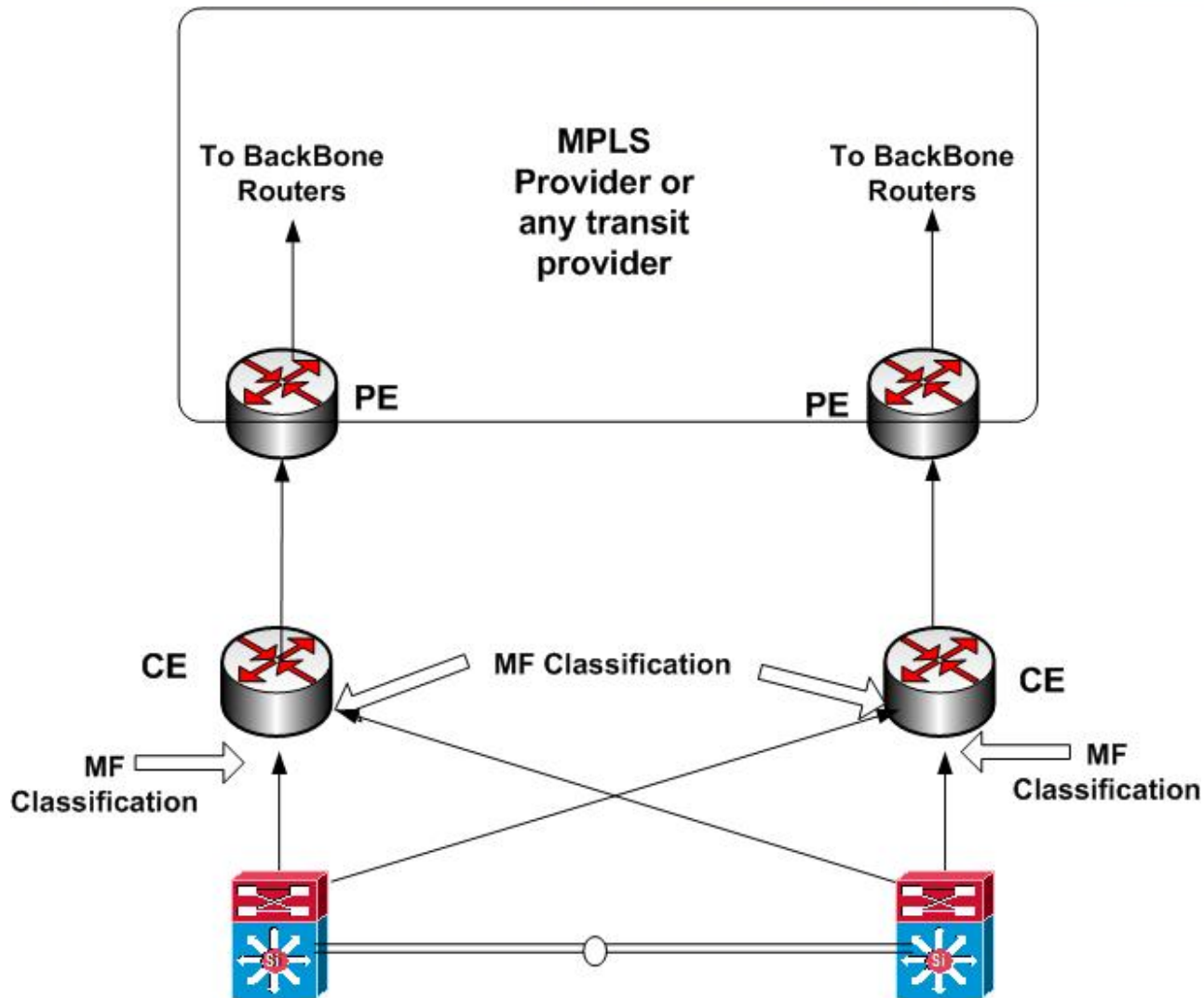


MF Classification on LAN Switching

- To trust or not to trust
- Decide where and what you want to classify
- LAN switches typically use port based classification
 - Permit tcp any any eq 80
 - Permit tcp any eq 80 any
 - Permit udp any any range 12000 13000
 - Permit udp any range 12000 13000 any
 - Permit udp any range 12000 13000 any range 12000 13000
- Port based classification is not ideal but may be sufficient



QoS Classification at the WAN Links



MF Classification on WAN links

- Congestion at the WAN links
- More Granular classification needed to minimize misclassification
- Port based classification not sufficient
- Granular classification prevents flows from gaming the system by getting into a better QoS class
- Precious WAN bandwidth is utilized more effectively
- Classification based on Layer 4 is imperfect!



MF Classification Granularity

```
term qos-App1 {
  from {
    source-address {
      10.32.0.0/24;
      172.24.1.0/25;
      192.168.0.0/20;
      192.168.28.0/22;
      192.168.32.0/19;
      192.168.64.0/18;
      192.168.128.0/17;
    }
    destination-address {
      10.11.0.1/32;
      10.32.0.2/32;
      172.16.1.0/24;
    }
    protocol udp;
    source-port 2001-3002;
    destination-port 2001-3002;
  }
  then {
    count CS4-app1;
    forwarding-class CS4;
  }
}
```



Juniper Commit Scripts

- Goal: Manage and deploy one generic MF classifier on all edge platform
- Challenges
 - 4 Queue Systems - M7i/M10i
 - 8 Queue Systems – J/M120/MX/M7i/10i with CFEB-E
 - Unique Routing-instances on a subset of routers
 - Unique Business logic/requirements
- Use commit script to modify forwarding-class and PLP to tailor it for 4 Queue vs 8 Queue platforms
- Use commit scripts to customize MF classifier for routers with routing instances



Juniper Commit Scripts

- Use commit script to customize shaping rate based on Bandwidth statement, not Port speed
- Use commit scripts look for bandwidth statement, interface tag, platform, and WAN provider and tailor policer/shaper
- Remove/modify policer/shaper if input values change
- Business logic encoded inside commit scripts



Policers (Cascading markdown)

- Traffic agreement with transit provider may require soft policers on your CE router
- E.g., If exceed 500 Mbps budget of CS4, then drop
- Place control back on customer (CE) side
 - Use a soft policer to mark down excess traffic over 500 Mbps to a lower class
 - Account and graph bps of markdown



Policers (Cascading markdown)

- **Example soft policer config**

```
set firewall family inet filter markdown interface-specific
set firewall family inet filter markdown term count-CS3-pre from forwarding-class CS3
set firewall family inet filter markdown term count-CS3-pre then count count-CS3-pre
set firewall family inet filter markdown term count-CS3-pre then next term
set firewall family inet filter markdown term police-CS3 from forwarding-class CS3
set firewall family inet filter markdown term police-CS3 then policer markdown-CS3
set firewall family inet filter markdown term police-CS3 then next term
set firewall family inet filter markdown term count-CS3 from forwarding-class CS3
set firewall family inet filter markdown term count-CS3 then count count-CS3-post
set firewall family inet filter markdown term count-CS3 then accept
```

- **Calculate BPS markdown**

$(\text{count-CS3-pre} - \text{count-CS3-post}) * 8 = \text{xxxx Mb/s marked down by policer}$

interval (sec)



Policers (Cascading markdown)

```
set firewall policer markdown-CS3 filter-specific
set firewall policer markdown-CS3 if-exceeding bandwidth-limit 500m
set firewall policer markdown-CS3 if-exceeding burst-size-limit 20250000
set firewall policer markdown-CS3 then loss-priority low
set firewall policer markdown-CS3 then forwarding-class CS2
```

```
set firewall policer markdown-CS2 filter-specific
set firewall policer markdown-CS2 if-exceeding bandwidth-limit 600m
set firewall policer markdown-CS2 if-exceeding burst-size-limit 20250000
set firewall policer markdown-CS2 then loss-priority high
set firewall policer markdown-CS2 then forwarding-class CS1
```



Operational aspects of QoS

- Requirements
 - Need to effect QoS changes globally in 10 mins
 - Need to account for traffic in each class
- Solution
 - Use Capirca Meta Language to generate Juniper and Cisco syntax and push ACLs to all edge devices
 - Use firewall counters to track byte and packet count
 - Use SNMP to poll counters and graph them
 - Counters provide alternative look at utilization graph compared to netflow exports

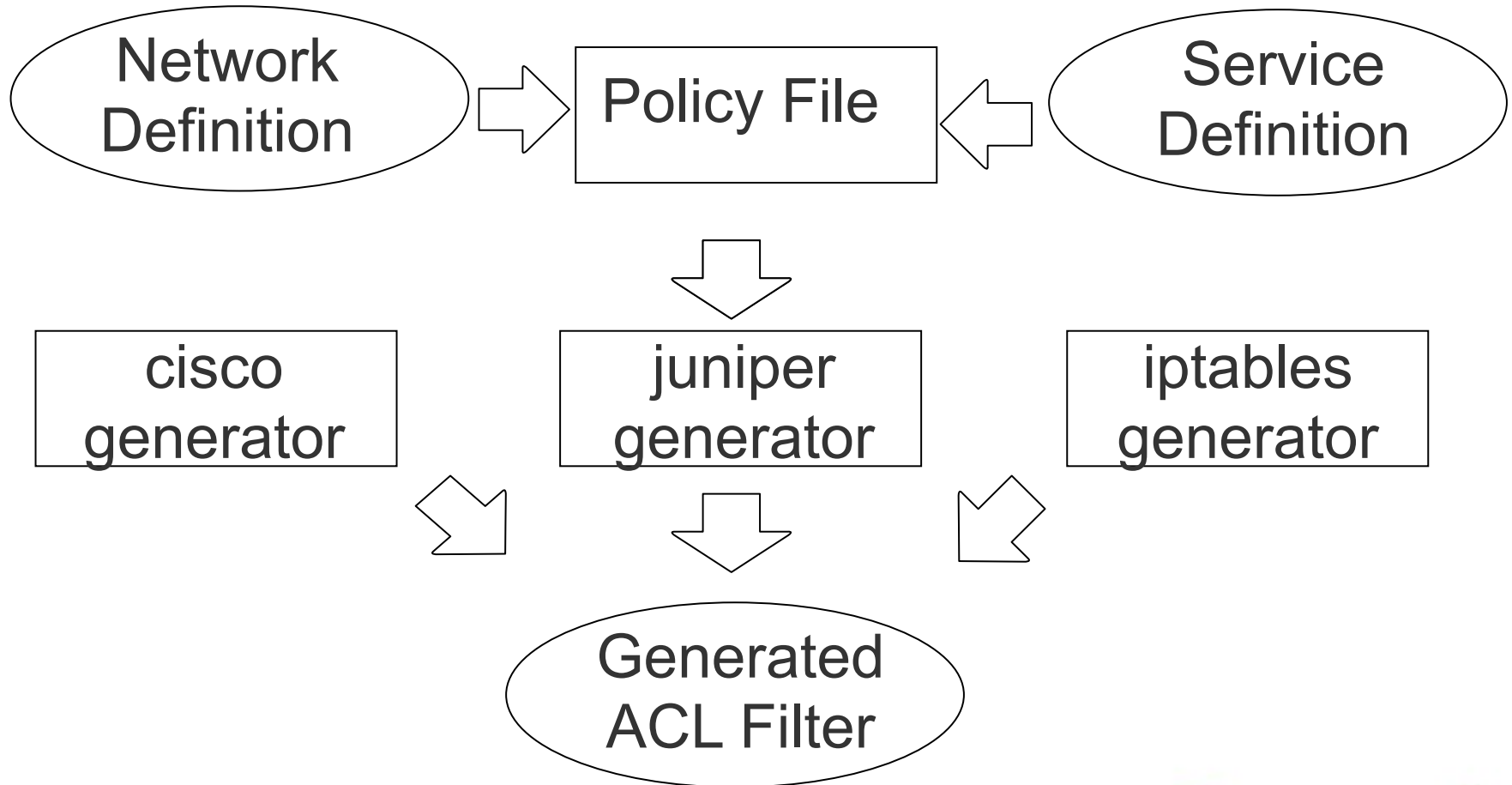


Capirca (<http://code.google.com/p/capirca/>)

- Built by Google Security team for ACL management
- Open sourced under Apache2 license in Sept 2009
- Same tool used to generate QoS MF classification ACL as security ACLs
- One policy file, generate many vendor specific ACLs (“Out of One, Many”)



ACL Generation Process (high level)



Network Definition File

```
RFC1918 = 10.0.0.0/8      # non-public
          172.16.0.0/12   # non-public
          192.168.0.0/16  # non-public
```

```
INTERNAL = RFC1918
```

```
LOOPBACK = 127.0.0.1/32 # loopback
           ::1/128       # ipv6 loopback
```

```
NYC_OFFICE = 100.1.1.0/24 # new york office
SFO_OFFICE = 100.2.2.0/24 # san francisco office
CHI_OFFICE = 100.3.3.0/24 # chicago office
```

```
OFFICES = NYC_OFFICE SFO_OFFICE
          CHI_OFFICE
```





Service Definition File

WHOIS = 43/udp

SSH = 22/tcp

TELNET = 23/tcp

SMTP = 25/tcp

VOICE = 12000-13000/udp

MAIL_SERVICES = SMTP
 ESMTP
 SMTP_SSL
 POP_SSL

DNS = 53/tcp 53/udp



Example Policy File

```
header {  
  comment:: "ACL for QoS classification"  
  target:: juniper qosFilter not-interface-specific  
}
```

```
term qos-Voice-CS4 {  
  comment:: "Voice Traffic"  
  source-address:: CORP_VOICE  
  destination-address:: CORP_VOICE  
  source-port:: VOICE  
  destination-port:: VOICE  
  protocol:: udp  
  counter:: CS4-Voice  
  qos:: CS4  
  policer:: markdown-CS4  
  action:: accept  
}
```



Generated Vendor ACL

```
firewall {
  family inet {
    replace:
    /*
    ** ACL for QoS classification"
    */
    filter qosFilter {
      /*
      ** Voice Traffic
      */
      term qos-Voice-CS4 {
        from {
          source-address {
            172.22.0.0/16;
          }
          destination-address {
            172.22.0.0/16;
          }
          protocol udp;
          source-port 1200-1300;
          destination-port 1200-1300;
        }
        then {
          count CS4-Voice;
          policer markdown-CS4;
          forwarding-class CS4;
          accept;
        }
      }
    }
  }
}
```



QoS over IPSec

- You will always have offices that need to IPSec back to headquarters due to circuit availability
- Use BGP as routing protocol to “glue” all offices together
- Juniper
 - Make sure DSCP bits are written to inner header and outer header
 - Juniper services PIC by default copies inner DSCP to outer
 - Rewrite rule necessary on inside of SP interface
 - This is slightly counterintuitive. “Inside or outside is from the perspective of the PFE, not the services PIC”
 - MF Classifier necessary on inside interface of SP if you run BGP over IPSec to learn routes; otherwise, BGP gets dumped into BE.
- Cisco
 - Use the “pre-classify” knob to classify flows before encryption



Best Practices

- Need to know your traffic pattern
 - How many Apps do you care to classify?
 - Move applications in higher QoS class to the top of the firewall filter
- Need to know your Platform
 - Juniper
 - Flows matching the last term of a complex 100 term firewall filters with firewall counters enabled will not yield Line rate Tput at 64 byte frames
 - Check to see if you have “info cell drops”
 - Cisco
 - Be aware of TCAM size limitations



Best Practices - Continue

- Test everything!
 - You need your own Qualification Lab/Testbed
 - You need to weigh in on cost vs performance
 - Work with vendor on bugs/feature requests before going live in production
- Monitor everything
 - Snmp poll appropriate QoS MIBs
 - Netflow
 - Know your usage pattern/trend



References

- Google Capirca <http://code.google.com/p/capirca/>
- Default Routing Engine Protocol Queue Assignment
<http://www.juniper.net/techpubs/software/junos/junos92/swconfig-cos/default-routing-engine-protocol-queue-assignments.html#id-10346810>
- Understanding ACL on Catalyst 6500
http://www.cisco.com/en/US/products/hw/switches/ps708/products_white_paper09186a00800c9470.shtml
- Configuring QoS for Virtual Private Networks
http://www.cisco.com/en/US/docs/ios/12_2/qos/configuration/guide/qcfvpn.html

