IP Telephony over Differentiated Services

Torsten Braun and Matthias Scheidegger
Computer Networks and Distributed Systems (Rechnernetze und Verteilte Systeme)
Institute of Computer Science and Applied Mathematics
University of Berne, Switzerland

www.iam.unibe.ch/~rvs

Dagstuhl, October 30, 2002
Overview

- Introduction
- Basic Concept
- Adaptive DiffServ Phone Implementation
- Performance Evaluation
- Conclusions
- References
Introduction

- Real-time applications are sensitive to QoS changes in the network.
- Network bandwidth and delay may be very dynamic.
- Applications can adapt themselves based on RTCP feedback.
- Differentiated Services (DiffServ) as scalable mechanism to provide QoS.
- Approach: Combination of adaptive applications and DiffServ
DiffServ Classes

- RFC 2474
  - ... PHBs selected by a Class Selector Codepoint SHOULD give packets a probability of timely forwarding that is not lower than that given to packets marked with a Class Selector codepoint of lower relative order ...

- Assumption: DiffServ classes can be ordered with respect to their QoS / price.
  - Price/quality of service offered by a higher class \( \geq \) price/quality of service offered by a lower class
Basic Concept

- Monitor QoS feedback information and always select cheapest class that provides sufficient QoS guarantees

- Application Scenarios
  - Charging is based on used DiffServ traffic
  - SLAs with multiple egress points
  - Congestion in access networks
Monitoring and Service Class Selection

- QoS parameters of DiffServ classes might vary → selection must be updated continuously
- Monitoring the used class and the next lower one might be sufficient, because service classes are ordered.
- Monitoring the next lower class by active probing using application-specific RTCP packets
  - Certain fraction of bandwidth for probing traffic (2 %)
  - Probing period needs to be long enough
  - Why not probing for next higher class?
    - Minimize network load
    - Switching to higher class does not decrease quality
- If current QoS is bad: switch to next higher class
- If current QoS is good and probing of next lower class shows good results: switch to next lower class
- Oscillations → random probing periods, hysteresis
FOREVER
    wait for new QoS monitoring
    create list L of requirements that cannot be met
    IF L is empty
        perform QoS monitoring for next lower class
        IF monitoring results are sufficient
            select next lower class
        ELSE
            keep current class
        ENDIF
    ELSE
        search next higher class with respect to L
        IF found
            select this class
        ELSE
            keep current class
        ENDIF
    ENDIF
ENDIF
LOOP
Adaptive DiffServ Phone Implementation

- DiffServ Phone (DSPhone) application
  - supports audio file I/O to support tests that can be reproduced
  - C++ implementation
- Supported DiffServ classes
  - Best-Effort
  - Assured Forwarding 1-4
  - Expedited Forwarding
- Probing based on application specific (APP) RTCP packets
- Ordering: BE \(\leq\) AF1 \(\leq\) ... \(\leq\) AF4 \(\leq\) EF
- Experiments in a Linux DiffServ test-bed
  (U Bern / NEC DiffServ implementation)
Performance Evaluation

Sender

Ingress node

Interior node

Receiver

PCM audio

Background traffic

Load

100 %
Switching between 2 Classes

![Diagram showing time and bandwidth with two classes of traffic]

- Interfering Traffic
- Service Class

Time (sec)

Bandwidth (Mbit/s)
Switching between 2 Classes
Switching between 3 Classes

![Diagram showing switching between 3 classes of traffic over time, with different bandwidth levels for each class.](image-url)
Switching between 3 Classes

Service Class
Loss Ratio
Loss Ratio, BE-Scout
Loss Ratio, AS1-Scout

Time (sec)

Class

Assured 2
Assured 1
Best Effort
Delay and Loss Rate

Router 2 overloaded
Router 3 overloaded

No DiffServ support for IP telephony application!
Conclusions

- Application reacts rather quickly and selects cheapest sufficient service class.
- Concept might be useful in several scenarios
- Congestion could be detected earlier by detecting increase of delay in order to avoid short loss peaks.
References


- UDPgen
  - time variant UDP traffic generator for experimentation networks
  - [www.iam.unibe.ch/~rvs/software](http://www.iam.unibe.ch/~rvs/software)