Special Topics: Diffserv Model

Xuan Chen
Nov 22, 2002

Outline
- Diffserv architecture
- Diffserv simulation in ns
- Implementation of diffserv model in ns
- Thanks for Nortel advanced network group for contributing the original code!

Diffserv (Differentiated Services)
- IP QoS architecture based on packet-marking
  - Differentiating traffic classes according to requirements (policies)
  - Discarding more packets in low priority traffic class upon congestion
  - Diffserv attempts to restrict complexity to only the edge routers
    - No end-to-end resource reservation

Diffserv Architecture
- Three major components
  - Policy and resource manager
    - Create network policies
    - Distribute policies to the Diffserv routers
  - Edge routers: packet marking
  - Core routers: PHB

Diffserv Policy
- A policy specifies which traffic receives a particular level of service in the network
- TSW (time sliding window) policy:
  - Traffic profile: expected throughput
  - Mark packets as IN when the measure traffic rate complies to its profile; Otherwise OUT
  - Drop more OUT packets upon congestion

Edge and Core Routers
- Edge router’s responsibilities:
  - Classifying incoming traffic according to policy specified and measurement
  - Marking packets with a code point that reflects the desired level of service
- Core router’s responsibilities:
  - Differentiating incoming packets based on code point and entries in PHB (per-hop-behavior) table
Outline

- Diffserv architecture
- Diffserv simulation in ns
- Implementation of diffserv model in ns

Steps for Simulation Configuration

- Setup edge and core “routers”
- Configure Diffserv queues
- Add diffserv policy
  - Entry in policy table
  - Entry in PHB table
- Collect packet statistics
- Example: token bucket marking policy with priority scheduling
  - More examples under ns/tcl/ex/diffserv/

Scenario

- CBR traffic from S1 and S2 to D
  - As we have discussed in previous session
- E1 and E2 are edge routers, C is core router
- Token bucket policy and priority scheduling

Configure Edge and Core Routers

For link (e1, c):

$ns simplex-link $e1 $core 10Mb 5ms dsRED/edge
$ns simplex-link $core $e1 10Mb 5ms dsRED/core

For link (e2, c):

$ns simplex-link $core $e2 5Mb 5ms dsRED/core
$ns simplex-link $e2 $core 5Mb 5ms dsRED/edge

Which to choose:
where does packet marking happen?

Diffserv Queue Configuration I

- Get handlers to diffserv queues
  set qE1C [[$ns link $e1 $core] queue]
  set qE2C [[$ns link $e2 $core] queue]
  set qCE1 [[$ns link $core $e1] queue]
  set qCE2 [[$ns link $core $e2] queue]

Diffserv Queue Configuration II

- Specify queue configurations
  $qE1C meanPktSize $packetSize
  $qE1C setNumQueues 1
  $qE1C setNumPrec 2
  ...
  $qE1C configQ 0 0 20 40 0.02
  $qE1C configQ 0 1 10 20 0.10
**Physical and Virtual Queues**

Packet 4 | 1 | 1
---|---|---
Packet 4 | 1 | 1

**Diffserv Queue Configuration III**
- Configure scheduling algorithms (default: RR)
- Configure priority scheduling
  - `$qCE2 setSchedularMode PRI`
  - `$qCE2 addQueueRate 0 3000000`
- $qCE2 meanPktSize $packetSize
- $qCE2 set numQueues_ 2
- $qCE2 setNumPrec 2

**Add Policy I**
- Add entries in policy table
  - `$qE1C addPolicyEntry [s1 id] [s dest id] TokenBucket 20 $cirO $cbsO`
  - `$qE1C addPolicyEntry [s2 id] [s dest id] TokenBucket 10 $cir1 $cbs1`
- Add Entries in policer table
  - `$qE1C addPolicerEntry TokenBucket 10 11`
  - `$qE1C addPolicerEntry TokenBucket 20 21`

**Add Policy II**
- Add Entries to PHB table
  - `$qE1C addPHBEntry 10 0 0`
  - `$qE1C addPHBEntry 11 0 1`
  - `$qE1C addPHBEntry 10 0 0`
  - `$qE1C addPHBEntry 21 0 1`
- Only PHB table is need for core router

**Collecting Statistics**
- Setup edge and core “routers”
- Configure Diffserv queues
- Add dffservlet policy
- Entry in policy table
- Entry in PHB table
- Collect packet statistics
- More examples under ~ns/tcl/ex/diffserv/

**Summary**
- Collect packets statistics.
- More examples under ~ns/tcl/ex/diffserv/
Outline

- Diffserv architecture
- Example and lab exercise
- Implementation of diffserv model in ns

Diffserv Model in ns

- Ported from Nortel
- An extension to ns
- Configuration in tcl: policy, edge and core routers
- Source code and sample scripts: under
  ~ns/diffserv and ~ns/tcl/ex/diffserv
- Add test suite and documentation
- Available since Dec 2000 or ns-2.1b8 release
- Widely used by ns users: “hot” topics in ns-users@isi.edu mailing list

Diffserv Model in ns: Revision

- Reorganizing the diffserv policy code to make adding new policies easier
- Providing new functions and bug fix
  - Added query functions for simulation statistics and queue states
  - Fixed bugs in priority scheduling
  - Thanks for ns-users’ contributions!

Implementing Diffserv Model in ns

- Classify traffic with physical and virtual queues
  - A code point in a packet is matched to a physical queue (traffic class) and a virtual queue (dropping preference)
  - Support different underline queuing disciplines (droptail, RED) and scheduling algorithms (round-robin, priority queue, etc)

Diffserv Queue I

- Implement Diffserv functionalities in queues
- Implement traffic classification with:
  - Modified RED queue: ds REDqueue
  - Contains up to 4 physical queue
  - Physical queue: traffic class
  - Real queue to hold packets
  - Contains up to 3 virtual queue
  - Virtual queue: drop preference
  - Individual RED parameters
  - Keep packet order among different virtual queues within one physical queue

Diffserv Queue II
**Edge and Core Routers**
- Implemented as edge-queue and core-queue
  - Derived from dsREDqueue
- Incoming packets:
  - At edge router (edge-queue): marked with code point
  - At core router (core-queue): queued at corresponding physical/virtual queue
- Outgoing packets:
  - Dequeued according to scheduling algorithms (among physical queues)

**Diffserv Policies in ns**
- Service profile at edge routers
  - Entry in policy and policer tables for source-destination pairs
  - Keep states for each pair
- PHB at core routers
  - Entry in PHB table: map code points to physical/virtual queues

**Policy Supported**
- TSW2CM and TSW3CM
- Token bucket
- Single rate three color marker
- Two rate three color marker

**Apply Policy**
- Edge routers keep the requirement and states for each source-destination pair in policy table
- Edge routers and core routers keep the relation: code point—traffic class/drop preference

**Policy Implementation in ns**
- Implement a supper class dsPolicy with virtual functions:
  - Meter: traffic measurement and state keeping
  - Policer: packet marking
- An actual policy is a child class derived from class Policy
  - Need to implement its own meter and policer functions
  - dumbPolicy: does nothing, but as an example
  - Edge routers refer to a certain policy by a pointer

**Steps to Add Customized Policy**
- “Register” your new policy in dsPolicy.h
- Define the new policy as a child class of class policy
- Write your own applyMeter and applyPolicer functions
- Add entries in functions addPolicyEntry and addPolicerEntry
- Example: DumbPolicy
Register New Policy

- Create identification

```c
#define DUMB 0
...
enum policerType {dumbPolicer, ...};
...
enum meterType {dumbMeter, ...};
```

Define new policy

```c
class DumbPolicy : public Policy {
public:
  DumbPolicy() : Policy() {
  void applyMeter(policyTableEntry *policy, Packet *pkt);
  int applyPolicer(policyTableEntry *policy, policerTableEntry *policer, Packet *pkt);
  ...}

  Write your own functions applyMeter and applyPolicer
```

Add Entries for New Policy

- Need to add entries for new policy in policy table, policer table, and functions to get statistics.

```c
void PolicyClassifier::addPolicyEntry(int argc, const char* const* argv) {
...
  if (strcmp(argv[4], "Dumb") == 0) {
    if (!policy_pool[DUMB])
      policy_pool[DUMB] = new DumbPolicy;
    policyTable[policyTableSize].policy_index = DUMB;
    policyTable[policyTableSize].policer = dumbPolicer;
    policyTable[policyTableSize].meter = dumbMeter;
  }
```

Example

- Modify dumbPolicy so that packets with even numbers are marked with lower priority.
- applyMeter: flow state keeping
- applyPolicer: packet marking based on flow state

Example---continued

- Question: what flow state should you keep?
- Try to work out this new policy as an optional task for lab...