QoS – Qualidade de Serviço

Luiz Arthur

O termo Qualidade de Serviço (**Quality of Service** – QoS) se **refere às garantias de desempenho estatístico que um sistema de rede pode dar com relação a perda, retardo, vazão e jitter**.

Dizemos que uma rede isócrona (redes que requerem tempos de transmissão iguais) que é **projetada para atender a limites de desempenho estritos** fornece garantias de QoS, enquanto uma **rede de comunicação de pacotes** que usa distribuição de melhor esforço **não fornece qualquer garantia de QoS**.

A **QoS** necessária **para transferência em tempo real de voz e vídeo pode ser utilizada em redes IP**? Se pode, com ela deve ser implementada? Uma importante controvérsia ronda as duas perguntas.

Por um lado, os engenheiros que projetam **o sistema de telefonia** insistem que a reprodução de voz de qualidade toll (qualidade de uma chamada telefônica convencional) **exige que o sistema subjacente forneça garantias de QoS** quanto a retardo e perda para cada chamada telefônica.

Por outro lado, os engenheiros que projetam o IP insistem que **a Internet funciona razoavelmente bem sem as garantias de QoS e que acrescentar QoS por fluxo é inviável porque os roteadores tornarão o sistema caro e lento**.

QoS – Qualidade de Serviço

A controvérsia do QoS gerou muitas propostas, implementações e experiências. **Embora a Internet funcione sem QoS**, **ela já costuma enviar dados que requerem tempo real**, tal como áudio e vídeo.

Tecnologias como ATM, que foram criadas a partir do modelo de sistema telefônico, fornecem garantias de QoS para cada conexão ou fluxo individual.

Para implementar QoS em redes baseados no protocolo TCP/IP surgiram duas propostas, uma é o conceito chamado Integrated Services (IntServ), que investigou a definição da **qualidade de serviço por fluxo**.

A outra proposta, adotada pelo **IETF** (Internet Engineering Task Force) é **um método Differentiated Services (DiffServ)** conservador. Esse esquema de serviço diferenciado, que sacrifica o controle fino para encaminhamento menos complexo, algumas vezes é **chamado de método de Classe de Serviço** (Class of Service – CoS).

QoS, utilização e capacidade

O debate sobre a **QoS** é remanescente dos debates anteriores sobre **alocação de recursos**, como aqueles travados sobre políticas de sistema operacional para alocação de memória e programação de processador.

QoS – Qualidade de Serviço

O problema central sobre o QoS é a utilização dos recursos:

- Quando uma rede possui **recursos suficientes** para todo o tráfego, às restrições da **QoS são desnecessárias**;

- Quando o **tráfego excede** a capacidade de rede, **nenhum sistema de QoS pode satisfazer às demandas de todos os usuários**.

Ou seja, uma rede com 1% de utilização não precisa de QoS, e uma rede com 101% de utilização falhará sob qualquer QoS.

Na verdade as pessoas que defendem QoS alegam que os mecanismos de QoS complexos atingem dois objetivos:

- Primeiro, dividindo os recursos existentes entre mais usuários, eles tornam o **sistema mais "justos"**.

- Segundo, **modelando o tráfego** de cada usuário, eles **permite**m que a **rede opere** em uma **taxa de utilização mais alta sem o risco de um colapso**.

QoS – Qualidade de Serviço

Um dos principais **argumentos contra os mecanismos de QoS complicados deriva das melhorias no desempenho das redes subjacentes**. A capacidade de rede aumentou dramaticamente.

À medida que aumentos rápidos na capacidade continuam, os **mecanismos de QoS meramente representam overhead desnecessário**.

Se a demanda por fluxo surgir mais rapidamente do que a capacidade, a QoS pode se tornar um problema econômico – associando preços maiores a níveis de serviço mais altos, **os provedores da Internet podem usar o custo para racionar a capacidade**.

Reserva de recursos do IntServ (RSVP)

Se a QoS for necessária, como uma rede IP poderá fornecê-la? Enquanto explorava **o IntServ**, o IETF **desenvolveu dois protocolos** para fornecer QoS: O Resource ReServation Protocol (**RSVP**) e o Common Open Policy Service (**COPS**). **Ambos exigem mudanças na infra-estrutura básica** – todos os roteadores precisam concordar em reservar recursos (por exemplo, largura de banda) para cada fluxo entre duas extremidades.

Existem dois aspectos para o IntServ ser implementado:

• Primeiro, antes que os dados sejam enviados, **as extremidades precisam enviar uma requisição que especifique os recursos necessários**, e **todos os roteadores** no caminho **precisam concordar** em fornecer os recursos; o procedimento pode ser visto como **uma forma de sinalização**.

• Segundo, à medida que os datagramas atravessam o fluxo, **os roteadores precisam monitorar e controlar o encaminhamento de tráfego**. O **monitoramento**, às vezes **chamado de policiamento de tráfego**, é necessário para garantir que o tráfego enviado em um fluxo não exceda os limites especificados.

O controle do enfileiramento e encaminhamento é necessário por duas razões: O roteador precisa **implementar uma política de enfileiramento** que atenda aos limites garantidos sobre retardo, e o roteador precisa **suavizar as rajadas de pacote**. Este último é, algumas vezes, chamado de modelagem de tráfego, e é necessário por que o tráfego de rede normalmente é feito em rajadas.

Por exemplo, um fluxo que especifica uma vazão média de 1Mbps pode ter 2 Mbps de tráfego para um milissegundo seguido de nenhum tráfego para um milissegundo. Um roteador pode remodelar a rajada enfileirando temporariamente datagramas que chegam e os enviando em uma taxa estável de 1Mbps.

QoS – Qualidade de Serviço

O RSVP cuida de requisições e respostas de reserva. Ele não é um protocolo de propagação de rota, nem impõe políticas quando um fluxo é estabelecido.

O RSVP opera antes que quaisquer dados sejam enviados.

Para iniciar um fluxo ponto a ponto, uma extremidade primeiro envia uma mensagem caminho RSVP de modo a determinar o caminho até o destino; o datagrama transportando a mensagem usa a opção roteador alerta para garantir que os roteadores examinem a mensagem.

Após receber uma resposta à sua mensagem caminho, a extremidade envia uma mensagem requisição para reservar recursos para o fluxo.

A requisição especifica os limites de QoS desejados; Cada roteador que encaminha a requisição para o destino precisa concordar em reservar os recursos que a requisição especifica.

Se qualquer roteador no caminho negar a requisição, o roteador usa o RSVP para enviar uma resposta negativa de volta à origem.

Se todos os sistemas ao longo do caminho concordarem em atender à requisição, o RSVP retorna uma resposta positiva.

QoS – Qualidade de Serviço

Cada fluxo RSVP é simplex (isto é unidirecional). **Se uma aplicação exigir garantis de QoS nos dois sentidos, cada extremidade precisa usar RSVP para requisitar um fluxo**. Como o RSVP utiliza encaminhamento existente, não há qualquer garantia de que os dois fluxos atravessarão os mesmos roteadores, nem a aprovação de um fluxo em um dos sentidos implica a aprovação do fluxo no outro sentido.

Resumidamente uma extremidade usa RSVP para requisitar um fluxo simplex através de uma rede IP com limites de QoS especificados. Se os roteadores no caminho concordarem em aceitar a requisição, eles aprovam; caso contrário, eles a negam. Se uma aplicação precisar de QoS nos dois sentidos, cada extremidade precisa usar RSVP para requisitar um fluxo separado.

Imposição do IntServ COPS

Quando chega uma requisição RSVP, um roteador precisa considerar dois aspectos:

- Viabilidade ou seja, se o roteador possui os recursos necessários para satisfazer à requisição;
- Política ou seja, se a requisição se encaixa dentro das restrições políticas.

A viabilidade é uma decisão local, um roteador pode decidir como gerenciar a largura de banda, a memória e a capacidade de processamento disponível localmente. Entretanto, a **imposição de política exige cooperação global** e todos os roteadores precisam concordar com o mesmo conjunto de políticas.

Para **implementar políticas globais**, a arquitetura IETF usa um modelo de **dois níveis**, com interação cliente-servidor entre os níveis.

Quando um roteador recebe uma requisição RSVP, ele se torna um cliente que **consulta um servidor conhecido como ponto de decisão política** (Policy Decision Point – PDP) para determinar se a requisição atende a restrições políticas.

O PDP não trata o tráfego; ele meramente avalia requisições para ver se elas satisfazem a políticas globais.

Se o PDP aprovar uma requisição, o roteador precisa atuar como um ponto de aplicação de política (Policy Enforcement Point – PEP) para garantir que o tráfego não exceda a política aprovada.

QoS – Qualidade de Serviço

O protocolo COPS define a interação cliente/servidor entre um roteador e um PDP (ou entre um foteador e um PDP local se a organização tiver múltiplos níveis servidores de política).

Embora o COPS defina se próprio cabeçalho da mensagem, o formato básico compartilha muitos detalhes com o RSVP. Em especial, o COPS usa o mesmo formato do RSVP para itens individuais em uma mensagem requisição.

Portanto, quando um roteador recebe uma requisição RSVP, ele pode extrair itens relacionados à política, colocá-los em uma mensagem COPS e enviar o resultado para um PDP.

DiffServ e comportamento por salto

O DiffServ se difere do IntServ de duas maneiras significantes:

- Primeiro, em vez de fluxos individuais, o DiffServ aloca serviço para uma classe (ou seja, um grupo de fluxos).

- Segundo, diferente do esquema RSVP, em que uma reserva é feita de fim-a-fim, o DiffServ permite que cada nó no caminho defina o serviço que uma determinada classe receberá.

Por exemplo, um roteador pode escolher dividir largura de banda de modo que a classe DiffServ conhecida como Expedited Forwarding (EF) receba 50% da largura de banda e os 50% restantes sejam divididos entre as classes que são conhecidas como Assured Forwarding (AF). Entretanto, o próximo roteador no caminho pode escolher dar à classe EF 90% de banda e dividir as classes AF entre os 10% restantes.

Usamos a frase comportamento por salto para descrever o método e para enfatizar que o DiffSerf não fornece garantias fim-a-fim.

Escalonamento de tráfego

Para implementar qualquer forma de QoS, um roteador precisa atribuir prioridades para o tráfego que sai e escolher que pacote enviar em um dado momento.

O processo de selecionar entre um conjunto de pacotes é conhecido como Escalonamento de tráfego, e o mecanismo é chamado de scheduler de tráfego.

QoS – Qualidade de Serviço

Existem quatro aspectos a serem considerados ao construir um scheduler:

- **Justiça**: O scheduler deve garantir que os recursos (ou seja, largura de banda) consumidos por um fluxo estejam dentro da quantidade atribuída para o fluxo;
- **Retardo**: Os pacotes em um determinado fluxo não devem ser retardados excessivamente;
- **Adaptatividade**: Se um determinado fluxo não tiver pacotes para enviar, o scheduler deve dividir a largura de banda extra entre outros fluxos proporcionalmente aos seus recursos atribuídos;
- **Overhead computacional**: Como opera no caminho rápido, um scheduler não precisa incorrer em muito overhead computacional.

O esquema de programação de tráfego prático mais simples chama-se **Weighted Round Robing** (WRR), pois **atribui um peso** a cada fluxo e tenta enviar dados do fluxo de acordo com o peso do fluxo.

Por exemplo, podemos imaginar três fluxos, A, B e C, com pesos 2, 2 e 4, respectivamente. Se todos os três fluxos tiverem pacotes esperando para serem enviados o scheduler deve enviar através do fluxo C (peso 4) o dobro do que através do fluxo A ou B (cada um com peso 2).

QoS – Qualidade de Serviço

Para parecer um scheduler WRR poderia atingir os pesos desejados selecionando dos fluxos na seguinte ordem

CCAB

Ou seja, o scheduler faz seleções repetidamente

CCABCCABCCAB...

O padrão parece alcanças os pesos desejados porque metade das seleções vem do fluxo C, um quarto vem de B e um quarto bem de A. Além, disso o padrão serve cada fila em intervalos regulares durante toda a sequência, significando que nenhum fluxo é retardado desnecessariamente (isto é, a frequência com que os pacotes são enviados de um determinado fluxo é constante).

Embora a sequência anterior realmente faça a taxa de pacote corresponder aos pesos atribuídos, **o método WRR não atinge o objetivo de fazer a taxa de dados corresponder aos pesos, pois os datagramas não possuem um tamanho uniforme**.

Por exemplo, se o tamanho médio dos datagramas no fluxo C é metade do tamanho médio dos datagramas no fluxo A (peso 2), se lecionar o fluxo C (peso 4) com o dobro da frequência do fluxo A tornará a taxa de dados dos dois fluxos iguais.

Para resolver o problema foi criado um algoritmo modificado que acomoda os pacotes de tamanho variável. Conhecido como **Deficit Round Robin** (DRR), o algoritmo **calcula os pesos em função dos bytes totais enviados**, e não pelo número de pacotes.

Inicialmente, o algoritmo DRR aloca um número de bytes para cada fluxo proporcionalmente à largura de banda que o fluxo deve receber. Quando um fluxo é selecionado, o DRR transmite o máximo de pacotes possível sem exceder o número predeterminado de bytes. O algoritmo, então, calcula o resto (ou seja, a diferença entre o número de bytes que foi alocado e o tamanho dos pacotes efetivamente enviados) e adiciona o resto à quantidade que será enviada na próxima vez.

Assim, o DRR mantém um total atualizado do “déficit” que cada fluxo deve receber. Mesmo que o déficit obtido em uma determinada rodada seja pequeno, o valor crescerá através da várias rodadas até que seja grande o bastante para acomodar um pacote extra. Portanto, **ao longo do tempo, a proporção dos dados que o DRR envia de um determinado fluxo atinge o valor ponderado para o fluxo**.

Algoritmos de programação roudin robin, como o WRR e o DRR, apresentam vantagens e desvantagens. A principal **vantagem** – e a razão de sua popularidade – surge da eficiência: uma vez que os pesos tenham sido atribuídos, **pouca computação é necessária** para fazer uma seleção de pacote. Na verdade, se todos os pacotes forem do mesmo tamanho e os pesos forem selecionados como múltiplos, a seleção ponderada pode ser obtida através do uso de um array, em vez de computação.

Apesar de sua popularidade, os algoritmos round robin possuem **desvantagens**. Primeiro, o **retardo** que um determinado fluxo experimenta **depende do número dos outros fluxos** que possuem tráfego para enviar. Na pior hipótese, um determinado fluxo pode precisar esperar enquanto o sheduler envia um ou mais pacotes de cada um dos outros fluxos. Segundo, como enviam uma rajada de uma determinada fila e, depois, retardam enquanto servem outras filas, os **algoritmos round roubin podem produzir jitter**.

Policiamento de tráfego

Um sistema que implementa programação de tráfego também precisa de um policial de tráfego que possa verificar se o tráfego que chega não excede um perfil estatístico declarado.

QoS – Qualidade de Serviço

Para ver por que esse mecanismo é necessário, considere um sistema em que um scheduler aloca 25% de largura de banda de saída para uma classe DiffServ, *Q*. Se três fluxos de entrada mapearem para a classe *Q*, os fluxos competirão pela largura de banda alocada para *Q*. **Portanto, o sistema precisa policiar cada fluxo de entrada para garantir que nenhum deles receba mais do que sua justa fatia.**

Vários mecanismos foram propostos para o policiamento de tráfego. Um dos primeiros, baseado no **método leaky bucket**, **usa um contador para controlar a taxa de pacote**. Periodicamente, o algoritmo incrementa o contador; cada vez que um pacote chega, o algoritmo diminui o contator. Se o contador se tornar negativo, significa que o fluxo que entra excedeu sua taxa de pacotes alocada.

Como foi dito, usar uma taxa de pacote fixa não faz sentido na Internet porque os datagramas variam de tamanho. Portanto esquemas de policiamento mais sofisticados foram propostos para acomodar pacotes de tamanho variável. Por exemplo, um mecanismo de **token bucket estende o método descrito antes fazendo o contador corresponder aos bits em vez de aos pacotes**. O contador é aumentado periodicamente de acordo com a taxa de dados desejada, e diminuído pelo tamanho de cada pacote que chega.

QoS – Qualidade de Serviço

Na prática, os mecanismos de policiamento descritos não exigem um timer para atualizar periodicamente um contador. Em vez disso, cada vez que um pacote chega, o policial examina o relógio para determinar quanto tempo decorreu desde a última vez que o fluxo foi processado, e usa o período de tempo para calcular um incremento para o contador.

QoS – Qualidade de Serviço

Fim

Leia livros sobre este assunto

QoS – Qualidade de Serviço

Fim

Leia livros sobre este assunto

QoS – Qualidade de Serviço

Fim

Leia livros sobre este assunto

Fim

Leia livros sobre este assunto