Trabalho Prático Nº2 – Proxy TCP reverso com monitorização proactiva

Duração: 5 aulas (não consecutivas, ver calendário)

Motivação

Para muito serviços, um único servidor não é suficiente para dar vazão aos pedidos dos clientes. Nesses casos é necessário ter uma pool de servidores com N servidores capazes de dar resposta aos pedidos. Ainda assim o serviço terá um único ponto de entrada para todos os clientes. Trata-se de um servidor de *front-end*, com um nome e um endereço IP únicos e bem conhecidos, cuja tarefa é atender as conexões dos clientes e desviá-las para um dos servidores de *back-end* disponíveis. Esse servidor designa-se normalmente por Proxy Reverso. Esta abordagem é comum nos serviços Web e está ilustrada na *figura 1*. A escolha do servidor pode ser cega, baseada por exemplo num algoritmo de Round-Robin, que faz uma distribuição equitativa das conexões pelos N servidores de back-end. Mas é possível fazer melhor, coligindo dados do estado do servidor e da rede, redirecionado em função de uma métrica dinâmica bem definida. Neste trabalho pretende-se desenhar e implementar um protótipo simples desta abordagem em duas fases: i) Recolha de informação dos servidores via UDP; ii) implementação de um font-end TCP que receba as conexões dos clientes, escolha um dos servidores disponíveis e intermedeie a conexão TCP para o servidor escolhido.

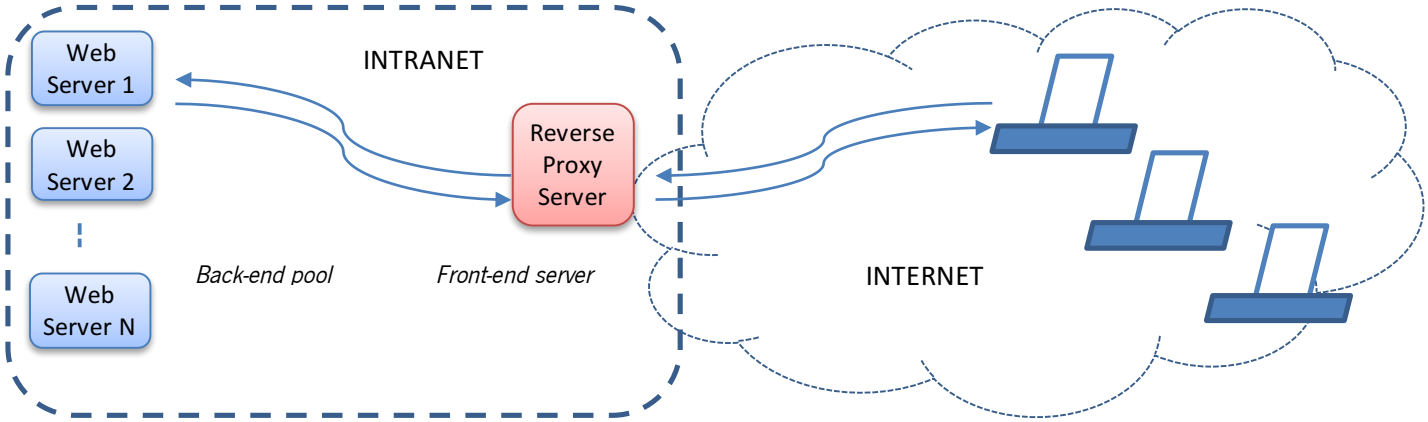


Figura 1

Objetivos

O principal objetivo deste trabalho é desenhar e implementar um serviço simples de proxy reverso TCP em que a escolha do servidor a usar se faz com base em parâmetros dinâmicos, como por exemplo o *RTT*, as *perdas* e *número de conexões TCP* do servidor. Será necessário numa primeira fase desenhar e implementar um protocolo sobre UDP para criar e manter atualizada uma tabela com dados/medidas recolhidas por servidor. Depois, numa segunda fase, pretende-se implementar um servidor proxy TCP genérico, que fique à escuta na porta 80 e redirecione automaticamente cada conexão TCP na porta 80 que receber para a mesma porta de um dos servidores disponíveis (o que aparente estar em melhores condições para o fazer).

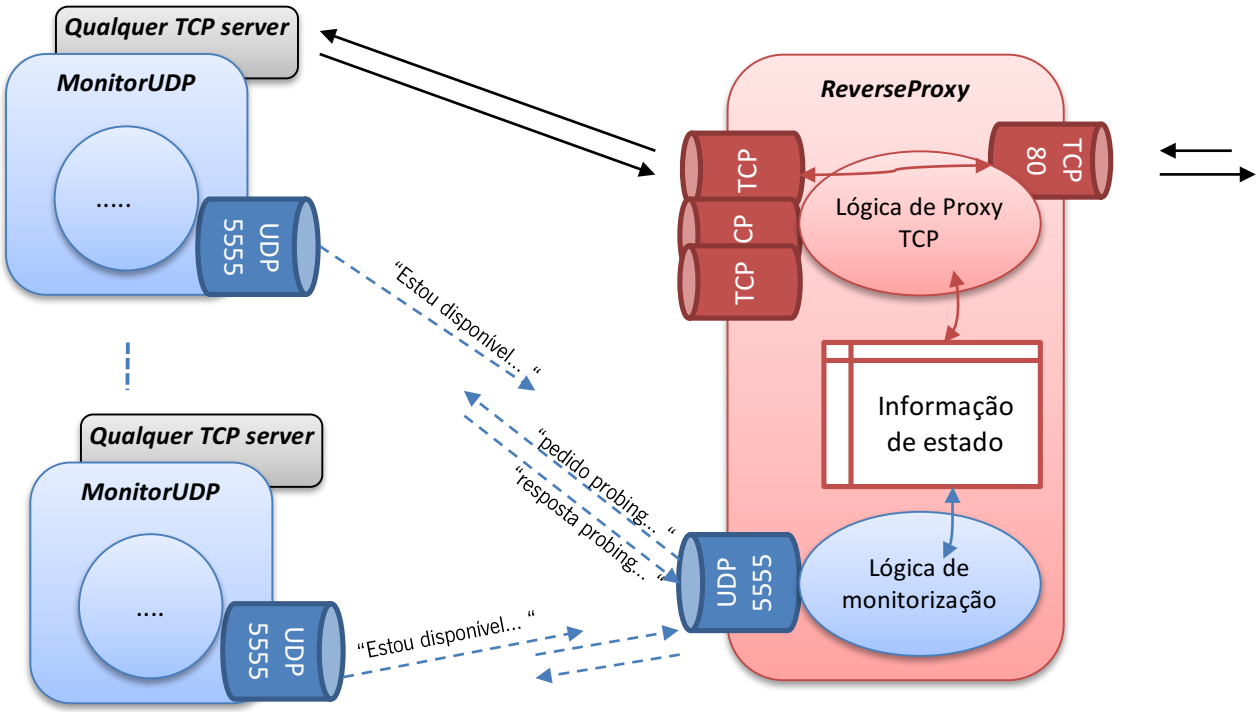
Descrição

O sistema a desenvolver pode ser separado basicamente em dois componentes: i) agentes de monitorização do estado dos servidores, a operar exclusivamente sobre UDP; e ii) servidor *reverse proxy*, capaz de intermediar as conexões TCP. A informação de estado deve ser coligida e atualizada regularmente numa tabela pelo agente de monitorização UDP. A informação recolhida na tabela será por sua vez usada pelo reverse proxy no algoritmo de escolha do servidor de back-end a usar. Sugere-se que na tabela conste por exemplo, o IP do servidor, o *round-trip time (RTT)* estimado entre o front-end e o servidor, a taxa de pacotes perdidos e o número de conexões TCP do servidor. Cabe ao servidor de *front-end* a responsabilidade de manter os dados atuais, fazendo um *pooling* periódico a cada servidor que se manifestou disponível. Cabe a cada servidor *back-end* manifestar-se disponível, periodicamente, ao servidor *front-end*.

O protocolo de monitorização UDP deve prever apenas dois tipos de interações. O registo periódico, feito com uma mensagem protocolar (PDU) bem definida para o efeito, e que não carece de qualquer confirmação no sentido contrário. E o pooling de estado, também periódico, que é uma interação do tipo pedido-resposta. Sempre que um agente receber um PDU com pedido de monitorização, deve enviar logo um PDU de resposta. As perdas são pois toleráveis e até se podem medir e guardar na tabela. A especificação dos formatos das mensagens (sintaxe e semântica) fica a cargo de cada grupo.

O servidor *reverse proxy* atua como mero intermediário. Recebe e aceita as conexões TCP dos clientes, escolhe um servidor disponível consultando a tabela de estado, e abre uma conexão TCP para o servidor escolhido. Tudo o que receber de um lado envia para o outro.

A *figura 2* mostra uma visão mais detalhada do sistema. O ***MonitorUDP*** é um agente de monitorização a instalar em cada servidor de back-end. Comunica exclusivamente pela porta 5555 UDP. O ***ReverseProxy*** é responsável por manter a tabela de estado, dialogando em UDP com os agentes ***MonitorUDP***. Ao mesmo tempo atende e intermedeia as conexões TCP dos clientes.



FASE 1: Protocolo de monitorização da pool de servidores (14 valores, 3 aulas)

Cada servidor da pool de servidores de back-end deverá ter um agente de monitorização instalado que podemos designar como *MonitorUDP*. Esse agente deve usar um protocolo de aplicação a funcionar sobre UDP, porta 5555. Mal o *MonitorUDP* entre em execução deve começar logo a enviar, de x em x segundos, uma mensagem de registo a informar o servidor principal (predefinido, passado como parâmetro) de que está disponível. Deve ainda responder a todas as mensagens com pedidos de monitorização que o servidor lhe enviar. Em todas as comunicações deve usar a porta 5555 como origem e como destino. É possível iniciar e terminar servidores a qualquer altura.

Etapas sugeridas para esta fase:

- Desenhar o PDU para registo junto do servidor central (esta interação é do tipo "indicação", só num sentido, sem resposta)
- Desenhar o(s) PDU(s) para monitorização do estado (esta interação é do tipo "pedido-resposta")
- Implementar agente ***MonitorUDP***
- Implementar apenas uma parte do servidor ***ReverseProxy***

FASE 2: Proxy TCP reverso (6 valores, 2 aulas)

O servidor deve atender pedidos TCP na porta 80. Depois de aceitar a conexão, decidir qual o melhor servidor disponível naquele instante consultando a tabela com informação de estado, e abrir uma nova conexão TCP com a porta 80 desse servidor. Depois funciona como mero intermediário entre estas duas conexões. Tudo o que receber do socket TCP do cliente é enviado para o socket TCP do servidor back-end escolhido e tudo o que receber no socket TCP do servicor back-end escolhido é enviado para o cliente. Funciona assim de modo totalmente transparente, qualquer que seja o cliente e o servidor envolvidos e qualquer que seja o protocolo de aplicação em uso. Sugere-se que o teste seja feito com clientes e servidores HTTP genéricos conforme descrito na secção "Cenário de Teste" que se segue.

Cenário de Teste

Pode-se usar como plataforma de teste o emulador CORE, embora tal não seja obrigatório. Para o cenário de teste mínimo são precisas três máquinas. Numa delas põe-se a correr o servidor **ReverseProxy**. Nas outras duas põe-se a correr o **MonitorUDP** e um servidor HTTP qualquer, capaz de servir páginas, como por exemplo o servidor web **mini-httpd** já usado no TP1. Pode-se assim usar um browser genérico para testar. Escrevendo no browser um URL com o endereço IP do ***ReverseProxy*** e porta 80, deverá ser possível obter como resposta uma página servida normalmente de um dos servidores HTTP de *back-end*.

Planeamento

O desenho e implementação deste serviço devem ser feitos ao longo das aulas práticas de CC, quer nas explicitamente dedicadas ao trabalho, como em todas as outras onde existir tempo livre. As fases de desenvolvimento são as seguintes:

FASE 1: As primeiras 3 aulas reservadas para o TP2
FASE 2: As últimas 2 aulas reservadas ao TP2

Relatório

O relatório deve ser escrito em formato de artigo com um máximo de 10 páginas (recomenda-se o uso do formato LNCS - *Lecture Notes in Computer Science*, instruções para autores em http://www.springer.com/computer/lncs?SGWID=0-164-6-793341-0). Deve descrever o essencial do desenho e implementação com a seguinte estrutura recomendada:

- Introdução
- Arquitetura da solução implementada
- Especificação do protocolo de monitorização
  * primitivas de comunicação
  * formato das mensagens protocolares (PDU)
  * interações
- Implementação
  * detalhes, parâmetros, bibliotecas de funções, etc.
- Testes e resultados
- Conclusões e trabalho futuro

Regras de submissão

Cada grupo de fazer a submissão (*upload*) do trabalho, usando a opção de "troca de ficheiros" associada ao seu grupo nos "Grupos" do Blackboard (*elearning.uminho.pt*), da seguinte forma:

- **CC-TP2-PLxx-Rel.pdf** para o relatório
- **CC-TP2-PLxx-Codigo.zip** ou **CC-TP2-PLxx-Codigo.rar** para a directoria com o código desenvolvido (devidamente documentado)

em que **xx** é o identificador de cada grupo (e.g. CC-TP2-PL21-Rel.pdf para o relatório TP2 do grupo PL21)