Redes de Dados Identificados por Nome

Redes de Computadores e Internet
2º Semestre 2024/2025
Projeto de Laboratório

1. Introdução

Uma *rede de dados identificados por nome* (*named data network*, NDN) é um sistema distribuído que tem por objetivo disponibilizar objetos de dados globalmente a todos os seus membros. Cada objeto está associado a um nome globalmente único. Cópias de um mesmo objeto podem estar distribuídas por nós distintos da rede. Um utilizador que pretenda obter determinado objeto lança uma *mensagem de interesse* na rede com o nome do objeto. Um nó que tenha armazenada uma cópia do objeto responde à mensagem de interesse com uma *mensagem de objeto* contendo o objeto e o seu nome, enviada sobre a mesma interface pela qual a mensagem de interesse foi recebida. Um nó que não tenha armazenada uma cópia do objeto reencaminha a mensagem de interesse para os seus vizinhos na rede. Um nó intermédio que receba uma mensagem de objeto reencaminha-a pela mesma interface pela qual recebeu anteriormente a correspondente mensagem de interesse, e armazena o objeto em cache. Ver Figura 2.

Fazem-se notar as seguintes características de uma rede NDN. Primeiro, os nomes dos objetos são identificadores NDN *anycast*, enquanto os nós nem necessitam de identificadores NDN. Segundo, cada objeto de dados é pesquisado através do seu nome, não havendo necessidade de um diretório que associe nomes a outros identificadores como no caso da internet e do DNS. Terceiro, os objetos viajam na rede em sentido inverso ao das mensagens de interesse e vão ficando disponíveis nos nós intermédios por onde passam.

A NDN implementada neste projeto será uma *rede de sobreposição* assente na internet. Na internet, cada nó é identificado por um endereço IP e um porto TCP de escuta. Cada adjacência na rede NDN é mapeada numa única sessão TCP entre dois nós da internet. Para simplificar, restringe-se a topologia de adjacências da rede NDN a uma árvore, isto é, tal que haja um caminho único de adjacências entre qualquer par de nós. Os nós precisam ser equipados com mecanismos que lhes permitam manter a topologia de adjacências em árvore quando um nó abandona a rede.

2. Manutenção da rede NDN

Na internet, cada nó é identificado pelo seu endereço IP seguido do seu porto TCP de escuta. Cada adjacência da rede NDN é concretizada numa única sessão TCP. Diz-se que um nó é *vizinho* de outro se com ele partilha uma adjacência. Para efeitos de manutenção da árvore aquando da saída de um nó, cada vizinho de um nó é classificado como *externo* ou *interno*. Esta classificação obedece a duas condições: se o nó Y é vizinho externo do nó X, então o nó X é vizinho interno do nó Y; numa rede com mais do que um nó, cada nó tem exatamente um vizinho externo. Destas duas condições infere-se que, numa rede com mais do que um nó, há exatamente dois nós que são vizinhos externos um do outro, portanto, também, vizinhos internos um do outro. Ver Figura 1.

Cada nó mantém a seguinte informação de estado: o identificador do seu vizinho externo; o identificador do vizinho externo do seu vizinho externo, ao qual chamamos *nó de salvaguarda*; os identificadores de cada um dos seus vizinhos internos. Um nó precisa saber o identificador do nó de salvaguarda para o caso de o seu vizinho externo sair da rede; precisa saber o identificador do seu vizinho externo para fornecer aos seus vizinhos internos um nó que lhes sirva de salvaguarda; e precisa saber os identificadores dos seus vizinhos internos porque há circunstâncias em que tem de escolher o seu vizinho externo de entre os seus vizinhos internos. (Note-se que os nós que são vizinhos externos um do outro, são salvaguarda de si próprios.) Há dois tipos de mensagens. Uma *mensagem de entrada* contém o identificador de um nó entrante. Uma *mensagem de salvaguarda* contém o identificador do vizinho externo de um nó, o qual virá a ser o nó de salvaguarda do destinatário da mensagem.

Entrada de um nó.

Um servidor de nós (fornecido pelo corpo docente) regista os identificadores de todos os nós atualmente pertencentes à rede. Um nó entrante na rede consulta o servidor de nós, o qual lhe fornece a lista de todos os nós registados na rede. Se a lista estiver vazia, então o nó regista-se no servidor de nós. Ele é o primeiro nó da rede. Contrariamente, se a lista não estiver vazia, então o nó entrante escolhe aleatoriamente um nó da lista, liga-se a este numa sessão TCP, estabelece-o como seu vizinho externo e envia-lhe uma mensagem de entrada. Posteriormente, o nó regista-se no servidor de nós. l

Saída de um nó.

Um nó que queira sair da rede apaga o seu registo no servidor de nós e remove todas as suas adjacências, fechando todas as sessões TCP que tinha em curso.

Remoção de uma adjacência.

Um nó que perca o seu vizinho externo e não seja salvaguarda de si próprio liga-se ao seu nó de salvaguarda numa sessão TCP, estabelece-o como seu vizinho externo, envia-lhe uma mensagem de entrada e envia a cada um dos seus vizinhos internos uma mensagem de salvaguarda. Por outro lado, se o nó for salvaguarda de si próprio e tiver pelo menos

um vizinho interno, então ele estabelece para vizinho externo um qualquer dos seus vizinhos internos, envia ao novo vizinho externo uma mensagem de entrada e envia a todos os seus vizinhos internos uma mensagem de salvaguarda.

Receção de uma mensagem de entrada.

Um nó que receba uma mensagem de entrada, estabelece o remetente da mensagem como vizinho interno. Se ele tiver vizinho externo, então responde ao remetente com uma mensagem de salvaguarda. Contrariamente, se ele não tiver vizinho externo, então estabelece o novo vizinho interno também como vizinho externo, envia-lhe uma mensagem de entrada e uma de salvaguarda.

Receção de uma mensagem de salvaguarda.

Um nó que receba uma mensagem de salvaguarda, atualiza o seu nó de salvaguarda.

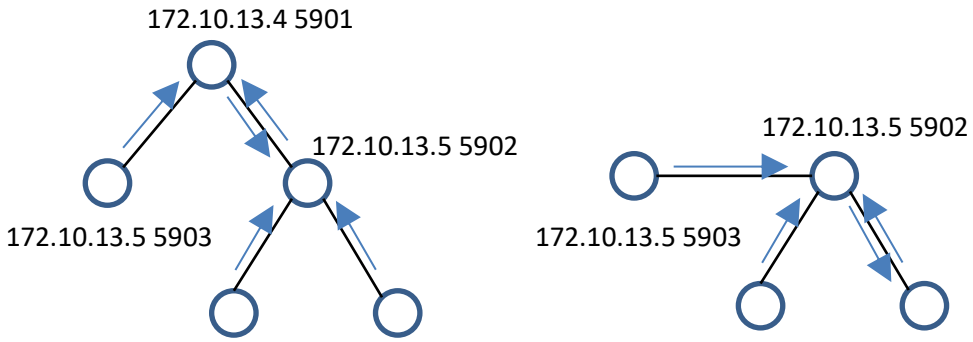

Figura 1. As linhas sólidas representam sessões TCP, enquanto as setas apontam no sentido dos vizinhos externos, O nó com identificador "172.10.13.4 5901" sai da rede. O nó com identificador "172.10.13.5 5903" liga-se ao seu nó de salvaguarda que é o nó com identificador "172.10.13.5 5902", e o nó com identificador "172.10.13.5 5902" escolhe um novo vizinho externo.

3. Obtenção de objetos na rede NDN

Na rede NDN, cada adjacência determina uma interface em cada um dos nós sobre os quais ela incide, não se fazendo distinção entre vizinhos internos e externos. Cada nó atribui um identificador distinto às suas interfaces de rede. Note-se que estes identificadores não têm de ser globalmente únicos. O utilizador de um nó cria objetos de dados que ficam armazenados no nó e que poderão posteriormente ser apagados por sua intervenção. Para simplificar a implementação, e sem perda de generalidade, omite-se os objetos e lida-se apenas com os seus nomes.

Há três tipos de mensagens. Uma *mensagem de interesse* procura um objeto de dados pelo seu nome. Uma *mensagem de objeto* contém um objeto procurado e o seu nome. Por último, uma *mensagem de ausência* indica que o objeto procurado não está presente. Cada nó mantém como informação de estado uma *tabela de interesses pendentes*. Cada entrada nesta tabela associa um nome procurado ao estado de cada uma das interfaces do nó

relativamente a essa procura. Há três tipos de estado. Uma interface no *estado resposta* é uma interface por onde uma mensagem de objeto ou de ausência deverá ser reencaminhada. Uma interface no *estado de espera* é uma interface por onde o nó reencaminhou uma mensagem de interesse, mas não recebeu ainda uma mensagem de retorno. Uma interface no *estado fechado* é uma interface por onde o nó recebeu uma mensagem de ausência. (Note-se que cada entrada numa tabela de interesses pendentes terá pelo menos uma interface no estado de espera.)

Início de uma pesquisa.

Suponha-se que um utilizador solicita um determinado objeto de dados. Se o nó não tiver esse objeto, ele envia uma mensagem de interesse por cada uma das suas interfaces e cria a correspondente entrada na tabela de interesses pendentes.

Receção de uma mensagem de interesse.

Suponha-se que a mensagem é recebida pela interface N.

1. Se o nó tiver o objeto, então envia a correspondente mensagem de objeto por N.
2. Se o nó não tiver o objeto nem entrada para o objeto na tabela de interesses pendentes, então: se N for a única interface do nó, ele envia uma mensagem de ausência por N; contrariamente, se N não for a única interface do nó, ele cria uma entrada para o objeto na tabela de interesses pendentes e reencaminha a mensagem de interesse por todas as suas interfaces exceto N.
3. Se o nó não tiver o objeto, mas já tiver uma entrada para o objeto na tabela de interesses pendentes, então N passa para o estado de resposta. Se, em resultado desta atualização, não houver interfaces no estado de espera, então é enviada uma mensagem de ausência por cada interface no estado de resposta.

Receção de uma mensagem de objeto.

Se o nome do objeto consta da tabela de interesses pendentes, então a mensagem é reencaminhada por todas as interfaces no estado de resposta, a entrada correspondente ao objeto é apagada da tabela de interesses pendentes e o objeto é guardado em cache. A cache terá um tamanho máximo e uma política de gestão.

Receção de uma mensagem de ausência.

Se o nome do objeto consta da tabela de interesses pendentes, então o estado da interface por onde a mensagem é recebida passa a fechado. Se, em resultado desta atualização, não houver interfaces no estado de espera, então é enviada uma mensagem de ausência por cada interface no estado de resposta.

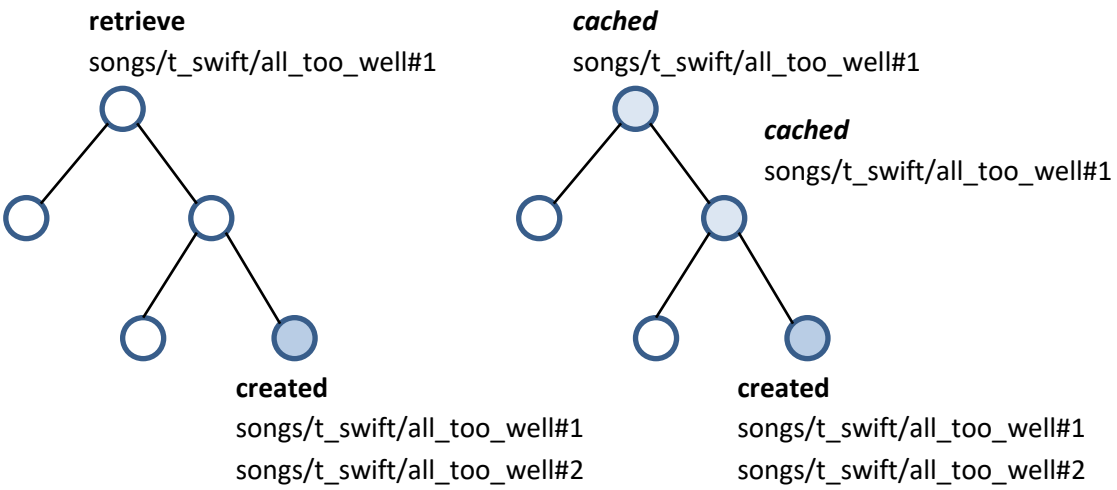

Figura 2.O nó em baixo à direita possui dois conteúdos. O nó em cima pede um dos conteúdos. Os nós no caminho entre o nó que pede o conteúdo e o nó que o possui guardam o conteúdo em cache.

4. Especificação

Cada grupo de dois alunos deve concretizar a aplicação **`ndn`** correspondente a um nó e composta pelos elementos seguintes:

- Comando de invocação de um nó;
- Interface de utilizador;
- Protocolos topológico e de registo;
- Protocolo NDN.

Nota: nos comandos e mensagens que se apresentarão de seguida, o espaço em branco e/ou o caráter ***<LF>*** (*line feed*) são separadores entre campos de mensagens, sendo que o caráter ***<LF>*** é também um separador entre mensagens enviadas sobre uma mesma sessão TCP. Por consequência, os nomes escolhidos pelo utilizador não devem conter espaços em branco ou o caráter ***<LF>***: deverão ser sequências de carateres alfanuméricos.

4.1 Comando de invocação da aplicação

A aplicação **`ndn`** é invocada com o comando

`./ndn` ***`cache IP TCP regIP regUDP`***

em que ***`cache`*** é um inteiro que representa o tamanho da cache, ***`IP`*** é o endereço IP da máquina que aloja a aplicação e ***`TCP`*** é o porto TCP de escuta. Em conjunto, IP e TCP formam o identificador do nó. Os parâmetros ***`regIP`*** e ***`regUDP`*** são o endereço IP e o porto UDP, respetivamente, do servidor de nós fornecido pelo corpo docente. Por omissão, tomam os valores **`193.136.138.142`** e **`59000`**.

4.2 Interface de utilizador

A interface de utilizador consiste nos comandos seguintes (abreviaturas entre parenteses).

- **`join (j)`** ***`net`***
  Entrada do nó na rede ***`net`***. Os valores de ***`net`*** são representados por três dígitos, podendo variar entre ***`000`*** e ***`999`***.
- **`direct join (dj)`** ***`connectIP connectTCP`***
  Entrada do nó ligando-se ao nó com identificador ***`connectIP connectTCP`***, sem registo no servidor de nós. Se ***`connectIP`*** for ***`0.0.0.0`***, então a rede é criada com apenas o nó.
- **`create (c)`** ***`name`***
  Criação de um objeto com nome ***`name`***. Os valores de ***`name`*** são representados por sequências alfanuméricas com um máximo de 100 carateres. Para simplificar, cria-se apenas o nome do objeto, omitindo-se o objeto propriamente dito.
- **`delete (dl)`** ***`name`***
  Remoção do objeto com nome ***`name`***.
- **`retrieve (r)`** ***`name`***
  Pesquisa do objeto com nome ***`name`***.
- **`show topology (st)`**
  Visualização dos identificadores dos vizinhos externo, de salvaguarda e internos.
- **`show names (sn)`**
  Visualização dos nomes de todos os objetos guardados no nó.
- **`show interest table (si)`**
  Visualização de todas as entradas da tabela de interesses pendentes.
- **`leave (l)`**
  Saída do nó da rede.
- **`exit (x)`**
  Fecho da aplicação.

4.3 Protocolo de registo

A comunicação com o servidor de nós efetua-se sobre UDP, consubstanciando-se nas seguintes mensagens.

- **`NODES`** ***`net`***
  O nó pede ao servidor de nós que lhe envie os identificadores de todos os nós presentes na rede ***`net`***.
- **`NODESLIST`** ***`net<LF>`***
  `IP1 TCP1<LF>`
  `IP2 TCP2<LF>`
  `…`
  O servidor de nós envia uma lista com uma linha por cada nó presente na rede ***`net`*** contendo o identificador desse nó.
- **`REG`** ***`net IP TCP`***

O nó regista-se na rede ***net***.

- **OKREG**
  O servidor de nós confirma o registo de um nó.
- **UNREG** ***net IP TCP***
  O nó remove o seu registo da rede ***net***.
- **OKUNREG**
  O servidor de nós confirma a remoção do registo de um nó.

4.4 Protocolo de topologia

As mensagens de entrada e de salvaguarda têm, respetivamente, os formatos seguintes.

- **ENTRY** ***IP TCP<LF>***
- **SAFE** ***IP TCP<LF>***

4.5 Protocolo NDN

As mensagens de interesse, objeto e ausência têm, respetivamente, os formatos seguintes.

- **INTEREST** ***name<LF>***
- **OBJECT** ***name<LF>***
- **NOOBJECT** ***name<LF>***

5. Desenvolvimento

Cada grupo de alunos deve adquirir a destreza necessária sobre programação em redes para realizar a aplicação proposta.

5.1 Etapas recomendadas

As etapas seguintes são recomendadas na concretização do projeto.

1. Leitura atenta do enunciado do projeto.
2. Criação de uma rede com vários nós com o comando **`direct join`**. Comando **`show topology`** operacional.
3. Criação de uma rede com vários nós com o comando **`join`**.
4. Saída de um nó da rede com o comado ***`leave`***.
5. Obtenção de um objeto com o comando ***`retrieve`***. Comandos **`create`**, **`delete`**, **`show names`** e **`show interest table`** operacionais.

5.2 Chamadas de sistema

O código da aplicação fará uso das seguintes primitivas:

- leitura de informação do utilizador para a aplicação: `fgets()`;
- conversão entre *strings* e tipos de dados: `sscanf()`, `sprintf()`;
- gestão de um cliente UDP: `socket()`, `close()`;

- gestão de um servidor UDP: `socket()`, `bind()`, `close()`;
- comunicação UDP: `sendto()`, `recvfrom()`;
- gestão de um cliente TCP: `socket()`, `connect()`, `close()`;
- gestão de um servidor TCP: `socket()`, `bind()`, `listen()`, `accept()`, `close()`;
- comunicação TCP: `write()`, `read()`;
- multiplexagem síncrona: `select()`.

5.3 Depuração

Faça uso das estratégias seguintes que permitem uma boa depuração do código:

- comente o código à medida que o desenvolve;
- use um depurador, `gdb` ou `xgdb`, para inspecionar a execução sequencial do código num nó;
- use o comando `nc` (`netcat`) para testar a comunicação com um cliente ou com um servidor, quer seja TCP ou UDP;
- execute o analisador de protocolos Wireshark para visualizar todas as mensagens trocadas com um nó e seus conteúdos;
- tente inter-operar um nó implementado pelo seu grupo com um nó implementado por outro grupo (isto é possível porque o formato das mensagens é o mesmo para todos).

Quer os clientes quer os servidores devem terminar graciosamente, pelo menos nas seguintes situações de falha:

- mensagens mal formatadas;
- sessão TCP fechada abruptamente;
- erro nas chamadas de sistema.

6. Entrega do Projeto

Cada grupo deve submeter no sistema fénix um ficheiro comprimido `.zip` (não é aceite `.rar`) com os três elementos seguintes: código fonte da aplicação **`ndn`**; a respetiva `makefile`; e uma ficha de auto-avaliação preenchida que será disponibilizada mais tarde. A execução do comando `make` deverá produzir o executável **`ndn`** sem quaisquer erros ou avisos. O código será testado apenas no ambiente de desenvolvimento do laboratório.

A data de submissão do projeto é sexta-feira, dia 28 de março às 23:59, ou segunda-feira, dia 31 de março às 23:59, esta segunda data e hora com dois valores de penalização. Recomenda-se vivamente que os alunos entreguem os seus projetos até ao dia 28 de março, sendo que a data de 31 de março acautela imprevistos.

7. Avaliação intermédia

Há uma avaliação intermédia na primeira aula da semana que começa a 10 de março e que vale três valores da nota final. A avaliação intermédia incidirá na capacidade de formar uma rede em árvore com os comandos **`direct join`** (etapa 2) e **`join`** (etapa 3), bem como na capacidade de atualizar a informação de estado necessária à manutenção da rede em árvore, a qual pode ser consultada com o comando **`show topology`** (etapa 2).