Sistemas Distribuídos

Pedro Brito

Sunday 24th May, 2020
18:29

1 Caso mais simples

Sincronização interna entre dois processos num sistema distribuído síncrono.

- São conhecidos os limites máximo (max) e mínimo (min) para o envio de mensagens, assim como para o desvio do relógio e para o tempo de execução dos processos.

Assumindo que o processo 1 envia uma mensagem ao processo 2 com o tempo que marca o relógio, t. Temos que :

- A incerteza no envio da mensagem será *u = (Max - Min*
- Se o processo 2 acerta o seu relógio para t + Max, o **máximo desvio** será também **u** porque a mensagem pode ter demorado Min.
- Mas se o processo 2 acertar o seu relógio para, t + (Max - Min) / 2 então o desvio entre os dois relógios será no máximo, (Max - Min) /2.

2 Como acertar um relógio ?

Obter UTC e corrigir o software do relógio.

2.1 Problemas

- O tempo nunca anda para trás.
- O valor lido do relógio físico deverá ser escalado pelo software de forma a ir atrasando lentamente, crescente.

2.2 Sistemas Assíncronos - Algoritmo de Cristian

- Obter UTC e corrigir o software to relógio.
- Calcular a estimativa para o tempo de propagação da mensagem
- p = (T1 - T0 - h) / 2
- Acertar o relógio do cliente para UTC + p
- Fazer várias medições para obter o valor de T1 - T0
  - Descartar valores acima de um determinado limite
  - Ou assumir os valores mínimos

2.3 Algoritmo probabilístico :

- a sincronização é conseguida se o RTT é pequeno quando comparado com a exatidão desejada
- a exactidão é tanto maior quanto o tempo de transmissão está perto do mínimo

Problema

- Ponto único de falha e congestionamento (bottleneck)

Solução

- Utilizar um conjunto de servidores com receptores de UTC
- O cliente faz o pedido em multicast para o conjunto de servidores e usa a primeira resposta que recebe

Problema

- Um servidor em falha ou malicioso pode provocar estragos.

Solução

- Autenticação
- Protocolo de acordo entre vários servidores que permita mascarar falhas.

3 Algoritmo de Berkeley (sincronização interna)

- É escolhido um computador para ser o co-ordenador (master)
- O master periodicamente contacta os outros computadores (slaves)
- O master faz uma estimativa do tempo local de cada slave,baseado no rtt.
- O master calcula o tempo médio de todos os computadores, ignorando valores de transmissão demasiado elevados e máquinas com tempos muito diferentes dos outros.
- Finalmente o master envia a cada computador o valor de que o seu relógio deve ser ajustado (esse valor pode ser positivo ou negativo)

Propriedas

- Precisão: depende do round trip time
- Ignora mensagens cujo tempo de transmissão é demasiado elevado.
- Que fazer se o master falha? Eleger um novo coordenador.

4 Modos de sincronização do NTP

5 Netowork Time Protocol (NTP)

- Múltiplos servidores de tempo espalhados pela Internet
- Servidores primários (ligados directamente aos receptores de UTC
- Servidores secundários sincronizam com os primários
- Servidores terciários sincronizam com secundários, etc
- Permite sincronizar um elevado número de máquinas
- Permite lidar com avarias de servidores Ou seja : Se um servidor secundário não consegue aceder a um primário, tenta aceder a outro. Existem servidores redundantes e caminhos redundantes entre servidores.
- Usa autenticação para verificar se a informação vem de fonte fiável

6 Modos de Sincronização do NTP

6.0.1 Modo "multicast"

- Usado em LANs de alta velocidade
- Um ou mais servidores faz periodicamente multicast do seu tempo para os outros servidores.
- Os receptores acertam os seus relógios assumindo um pequeno atraso de transmissão.

6.0.2 Modo "procedure call"

- Similar ao algoritmo de Cristian
- Clientes solicitam o tempo de um ou vários servidores, e estes enviam o valor do seu relógio.
- Adequado quando o multicast não está disponível

6.0.3 Modo "symmetric"

Maior exactidão, usado em servidores primários ou próximos

Para cada par de processos calcula-se um *offset*, **o**, que corresponde à diferença entre os dois relógios, e um delay, **d**, que é o tempo total de transmissão das duas mensagens.

Se o offset do relógio de A em relação ao de B for **o** (Tb = Ta + **o**) os tempos de transmissão de mensagens de **m** e **m'** forem **t** e **t'**.

Temos então que :

$$T_2 = T_1 + t + o \tag{1}$$

E

$$T_4 = T_3 + t' - o \tag{2}$$

Round Trip Delay

$$d_i = t + t' = T_2 - T_1 + T_4 - T_3 \tag{3}$$

Supondo que T2 - T1 = T4 - T3 então :

$$O = T_3 + ((T_2 - T_1) + (T_4 - T_3))/2 - T_4 = ((T_2 - T_1) + (T_3 - T_4))/2 \tag{4}$$

Se o relógio de A é mais rápido, O ¡ 0