Fault Tolerance Middleware for Cloud Computing

W. Zhao & P. Melliar-Smith & L. Moser

Sistemas Distribuídos e Tolerância a Falhas

Ivan Pires – m3797
Gilberto Melfe – m4088

Introdução

A “cloud computing” tem por objectivo fornecer serviços fiáveis, num ambiente de rede, sem que os utilizadores tenham necessidade de ter em conta detalhes de “hardware” e “software” subjacentes ao sistema.

Um dos desafios é garantir que esses serviços funcionam ininterruptamente.

O “middleware” “Low Latency Fault Tolerance” providencia tolerância a falhas a aplicações distribuídas como um serviço prestado.

Conceitos Básicos
Modelo de Faltas

O “middleware” LLFT replica processos das aplicações para as proteger contra vários tipos de falhas, em particular:

- “Crash fault” – em que um processo deixa de produzir resultados;
- “Timing fault” – em que um processo não produz o seu resultado dentro de uma janela temporal;

Conceitos Básicos
Tipos de Replicação

O “middleware” LLFT suporta dois tipos de replicação (“leader/follower”):

- “Semi-active” – em que o “primary” ordena as mensagens que recebe, realiza as operações correspondentes e fornece a ordenação para operações não determinísticas aos “backups”; e um “backup” recebe e faz “log” das mensagens recebidas, realiza as operações de acordo com a ordenação recebida e faz “log” das mensagens “enviadas” (mas não as envia);

Conceitos Básicos
Tipos de Replicação (Cont.)

O middleware LLFT suporta dois tipos de replicação (leader/follower):

- "Semi-passive" – em que o "primary" ordena as mensagens que recebe, realiza as operações correspondentes e fornece a ordenação para operações não determinísticas aos "backups"; Além disto o "primary" comunica actualizações ao seu estado (e a ficheiros e bases de dados) aos "backups", que recebem e fazem "log" das mensagens, actualizam o seu estado mas não realizam as operações e não produzem mensagens;

Conceitos Básicos
Service Groups e Process Groups

Um “process group” é um conjunto de réplicas de um processo, em que existe um “primary” e um ou mais “backups”.

Um “service group” é um conjunto de “process groups” que interagem entre si, e que em conjunto fornecem um serviço ao utilizador.

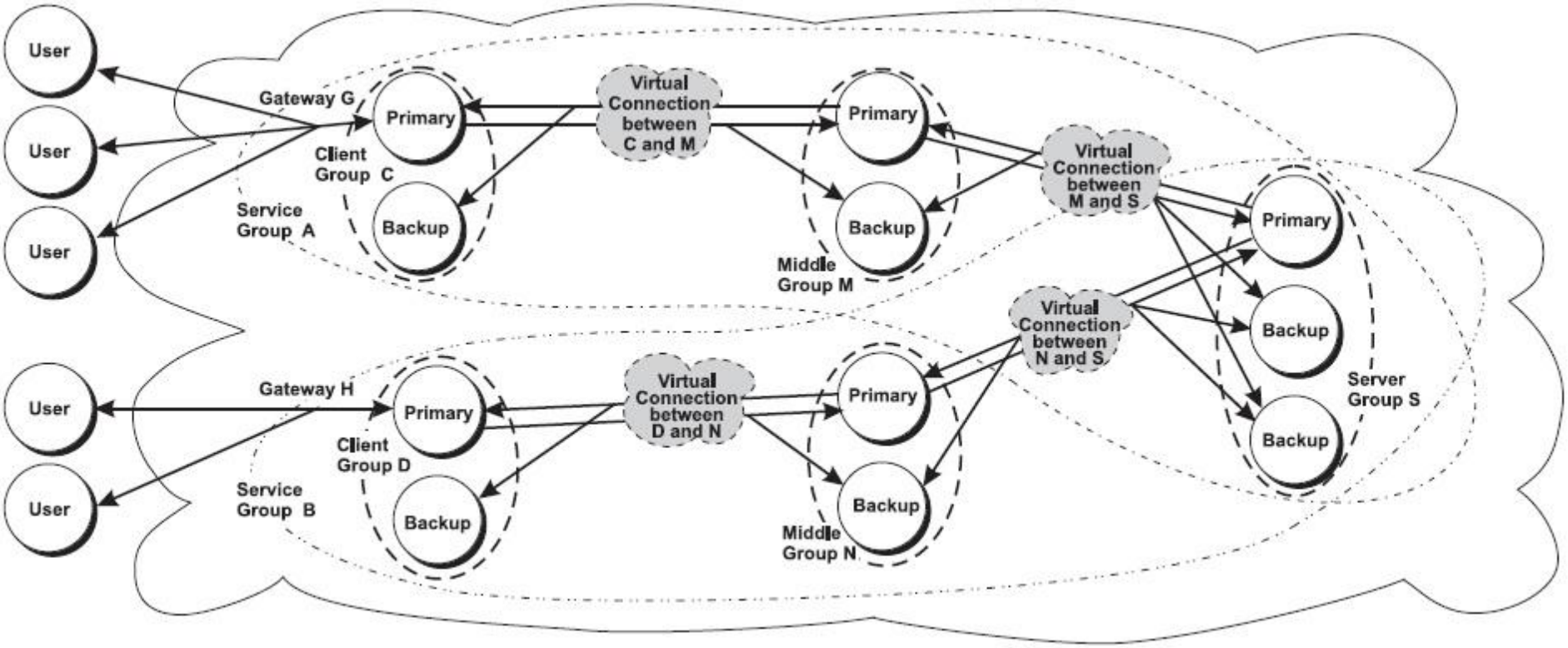
Um membro de um “process group” tem de conhecer o “primary” desse grupo.

Conceitos Básicos
Conexões Virtuais

Uma virtual connection é uma ligação (“full-duplex”, “many-to-many”) entre dois “process groups” através da qual estes trocam mensagens.

Cada um dos “process groups” é identificado por um “virtual port”, em que todos os membros do grupo estão à escuta.

Cada “service group” possui um porto através do qual o utilizador acede ao serviço fornecido (através de um “gateway”).


User
User
User
User
User
Gateway G
Client
Group C
Service
Group A
Primary
Backup
Virtual
Connection
between
C and M
Primary
Backup
Middle
Group M
Virtual
Connection
between
M and S
Primary
Backup
Backup
Server
Group S
Virtual
Connection
between
N and S
Gateway H
Client
Group D
Service
Group B
Primary
Backup
Virtual
Connection
between
D and N
Primary
Backup
Middle
Group N

Composição do “middleware” LLFT

Este “middleware” é composto de três partes fundamentais:

- o “Low Latency Messaging Protocol”;
- o “Leader-Determined Membership Protocol”;
- e a “Virtual Determinizer Framework”.

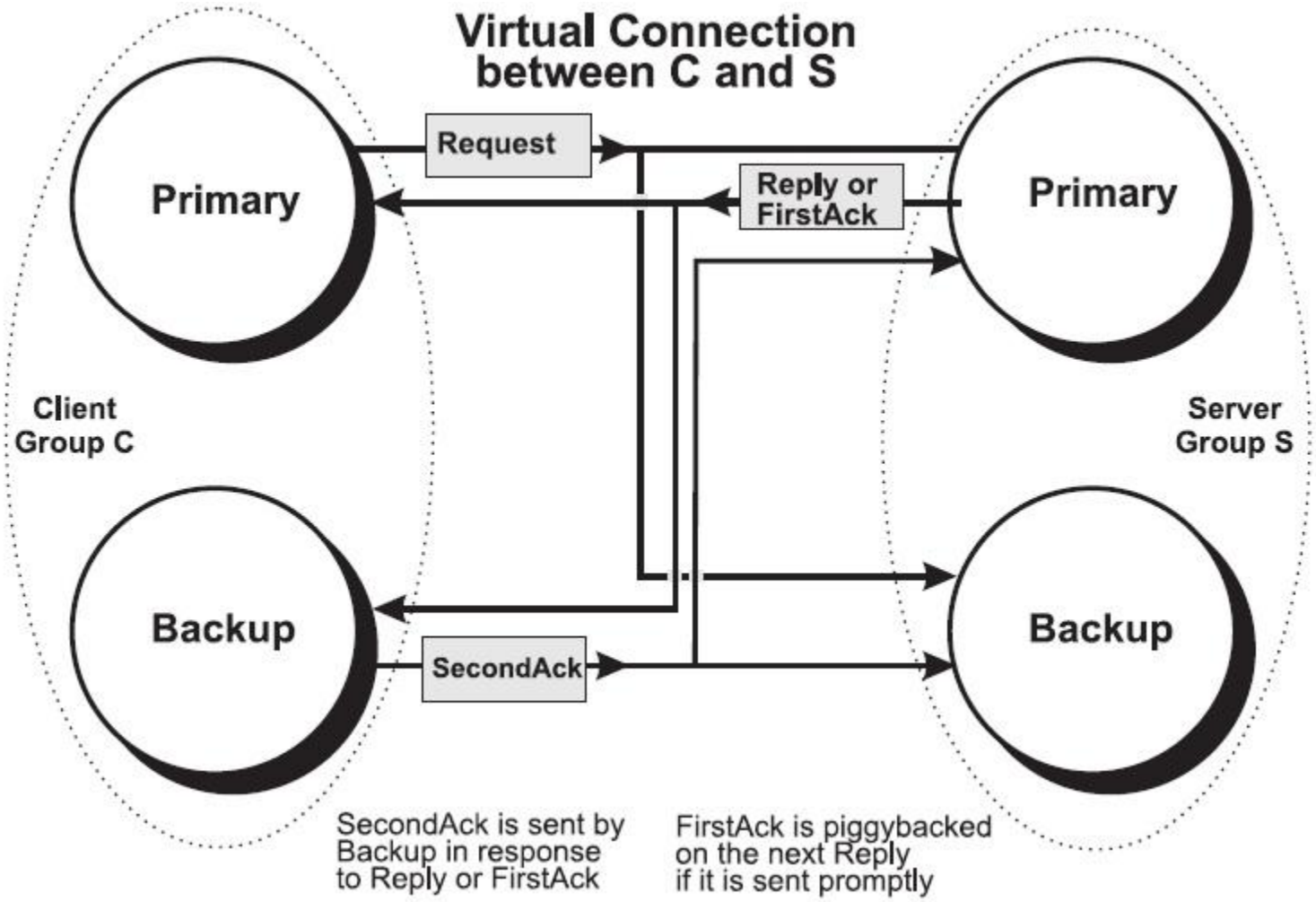
“LLFT Messaging Protocol”

Fornece um serviço de transferência de mensagens da aplicação, em “multicast”, garantindo:

- Entrega fiável;
- Ordenação total;

Usa como base o UDP “multicast”.

Incorpora mecanismos de controlo de fluxo semelhantes aos do TCP.


Virtual Connection
between C and S
Request
Reply or
FirstAck
Primary
Primary
Client
Group C
Server
Group S
Backup
Backup
SecondAck
SecondAck is sent by
Backup in response
to Reply or FirstAck
FirstAck is piggybacked
on the next Reply
if it is sent promptly

“LLFT Messaging Protocol”
Entrega Fiável

O “primary” num grupo origem (C) faz “multicast” das mensagens da aplicação para um grupo destino (S) numa conexão virtual.

Um “backup” no grupo C cria e faz log das mensagens geradas mas não as transmite.

O “primary” guarda as mensagens numa “Sent List” e reenvia-as caso não sejam “acknowledged” dentro de um certo tempo (“timeout”).

“LLFT Messaging Protocol” Entrega Fiável (Cont.)

O “primary” num grupo destino (S) inclui nas mensagens que envia ao grupo C o “sequence number” da última mensagem que recebeu em sequência, ou, se não tiver mensagens da aplicação para enviar, envia um “FirstAck” para indicar esse “sequence number”.

Um “backup” no grupo origem (C) confirma a recepção de um “FirstAck” com um “SecondAck”…

“LLFT Messaging Protocol” Ordenação Total

O “primary” num grupo (C) comunica a ordenação das mensagens aos “backups” do seu grupo, para que possam reproduzir as acções do “primary” e manter a consistência da replicação.

O “primary” (C) “piggybacks” em cada mensagem que envia a ordenação das mensagens que enviou/recebeu desde a última mensagem que enviou.

Nota: os “backups” no seu grupo não recebem essa ordenação directamente do “primary”.

“LLFT Messaging Protocol” Ordenação Total (Cont.)

O “primary” do grupo destino (S) reflecte a informação de ordenação para o grupo origem (C) na próxima mensagem que lhe envia.

O “primary” no grupo C inclui essa informação nas mensagens que envia até a receber de volta.

Isto funciona nos dois sentidos.

“LLFT Membership Protocol”

Este protocolo visa fornecer um método determinístico de “nomear” um “primary” dentro de um “process group”, bem como determinar quais os “backups” que fazem parte desse grupo, quando ocorre uma falha.

Segundo os autores este protocolo é mais rápido que outros do tipo “multi-round”.

Esta mudança na chamada “membership” é levada a cabo mantendo a consistência da replicação.

“LLFT Virtual Determinizer Framework”

As aplicações que funcionam em “cloud computing” envolvem diversas fontes de não-determinismo, que é preciso sanear/mascarar para conseguirmos manter “strong replica consistency”.

Vamos tornar a aplicação “virtuallly deterministic.”

O algoritmo proposto é genérico e foi instanciado para:

“Multi-threading”, comunicações via “Sockets” e operações dependentes do tempo.

Implementação

O “middleware” está implementado em C++, em Linux.

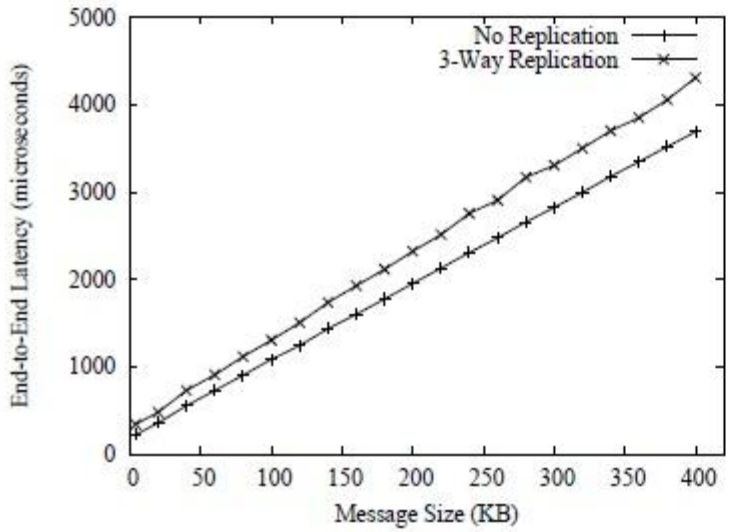
Usa a técnica de “library interpositioning” para controlar as interacções da aplicação com o sistema; é compilada como uma biblioteca partilhada e inserida no espaço de endereçamento da aplicação.

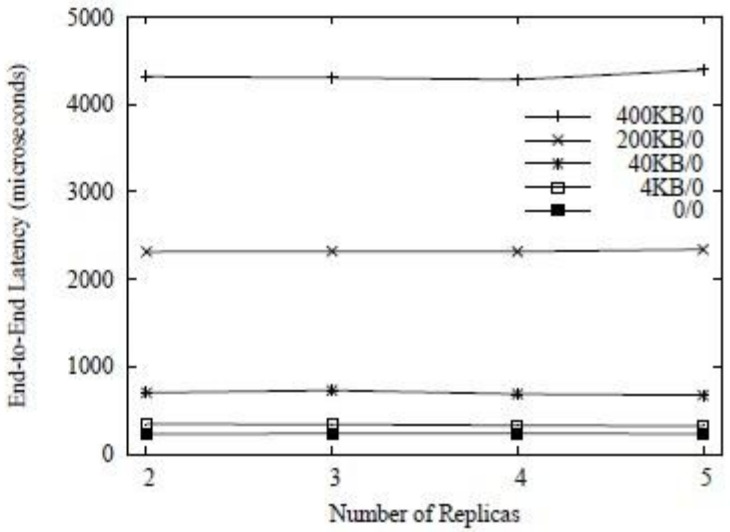
É assim transparente para a aplicação.

Desempenho

O “LLFT Messaging Protocol” incorre num “overhead” moderado.

Também é escalável à medida que o grau de replicação aumenta.





Conclusões

O “middleware” “Low Latency Fault Tolerance” fornece tolerância a falhas a aplicações distribuídas que funcionam num ambiente de “cloud computing”.

Com recurso a este “middleware”, aplicações com características adequadas podem ser replicadas com manutenção de “strong replica consistency”, sem modificação nas aplicações.

O desempenho demonstrado é favorável à sua utilização no ambiente destino (“cloud”).