Modelos Fundamentais

Carlos Ferraz
cagf@cin.ufpe.br

O que vimos até agora (I)

- História
  - Anos 60-70: sistemas centralizados (caros!)
  - Anos 80: computadores pessoais (acessórios caros e pouco aproveitáveis! – ex: impressora usada por pouco tempo)
  - Sistemas em rede: computadores independentes interligados através de rede
  - Sistemas distribuídos: computadores independentes interligados através de rede + transparência
- Definição: coleção de computadores/sistemas independentes que aparecem para os usuários do sistema como um único computador/sistema

O que vimos até agora (II)

- Desafios
  - Heterogeneidade, abertura, segurança, transparência, consistência, tolerância a falha
- Características
  - Compartilhamento de recursos, abertura, concorrência, escalabilidade, tolerância a falha, transparência
- Modelos arquiteturais
  - Cliente-servidor: maturidade
  - Peer-to-peer: autonomia dos pares
  - Objetos distribuídos: extensão do modelo cliente-servidor
    - Encapsulamento, reuso, *object brokers, object services (middleware)*
- Conceitos: plataforma, *middleware*, serviço

Modelos Fundamentais

- Modelos de interação
- Modelos de falhas
- Modelos de segurança

Por que ter modelos de sistemas?

- Clarear idéias
- Organizar e classificar abordagens diferentes
- Fazer generalizações

O que deve ser capturado em um modelo?

- Interação:
  - Como componentes interagem e se coordenam para resolver um problema?
- Falha:
  - Quais as falhas em potencial e como elas afetam os resultados?
- Segurança:
  - Quais as vulnerabilidades do sistema e como devem ser tratadas?

Modelo de Interação

- Processos interagem através de passagem de mensagem para resolver um problema
- Interação é influenciada por:
  - Desempenho de canais de comunicação
    - Latência
    - Banda-passante
    - *Jitter*
  - Sincronização de relógios

Relógios e Ordenação de Eventos

- Para indicar ordenação, em geral mensagens são marcadas com tempo (*time-stamped*)
- *Time-stamps* (selos de tempo) são normalmente baseados em valores do relógio local
- Relógios locais não são sincronizados
  - Como podem ser?
- Não há um relógio global

 7

Sincronização

SDs Síncronos: Podem limitar

SDs Assíncronos: Sem limites

- Tempo para cada passo de processamento
- Tempo para transmitir cada mensagem
- Atraso de relógio para cada processo

 8

Tempo lógico

- Para um único processo, eventos são ordenados pelo relógio (físico) local
- Em um SD, em geral não se pode usar relógios físicos, por conta de não serem perfeitamente sincronizados
- A ordem de eventos que ocorrem em diferentes processos de um SD pode ser crítica

 9

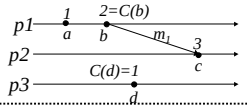
Ordenação

- Dois pontos óbvios:
  - Se 2 eventos ocorreram em um mesmo processo, então ocorreram na ordem em que são observados
  - Quando uma mensagem é enviada entre processos, o evento de envio da mensagem ocorreu antes do evento de recepção da mensagem
- Generalizando, Lamport (1978) estabeleceu a relação *aconteceu-antes*

 10

Formalização da relação *aconteceu-antes*

- Se $\exists$ processo $p : x \rightarrow^{p} y$, então $x \rightarrow y$
  - $x \rightarrow^{p} y$: se os eventos x e y ocorrem em um mesmo processo p, e x ocorre antes de y
- $\forall$ mensagem m, $send(m) \rightarrow rcv(m)$
- Se x, y e z são eventos, tais que $x \rightarrow y$ e $y \rightarrow z$, então $x \rightarrow z$
- Seja $C(a)$ o *timestamp* de um evento a
  - $C(b) > C(d) \not\Rightarrow d \rightarrow b$



 11

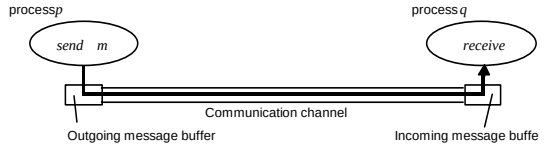
Modelo de Falhas

Define os modos como falhas podem ocorrer para entender os efeitos de falhas

- Falhas de omissão: processo ou canal falha para fazer alguma coisa
- Falhas arbitrárias: normalmente se referem a sabotagens
- Falhas de tempo: limites de tempo excedidos

 12

Processos e canais



Instructor's Guide for Coulouris, Dollimore and Kindberg Distributed Systems: Concepts and Design Edn. 3
© Addison-Wesley Publishers 2000

Falhas de omissão e arbitrárias

Classe de falha	*Afeta*	*Descrição*
Fail-stop	Processo	Processo para e permanece parado. Outros processos podem detectar este estado.
Crash	Processo	Processo para e permanece parado. Outros processos podem não detectar este estado.
Omission	Canal	Uma mensagem inserida em um buffer de saída nunca chega no buffer de entrada do outro lado.
Send-omission	Processo	Um processo completa um *send*, mas a mensagem não é colocada no buffer de saída.
Receive-omission	Processo	Uma mensagem é colocada no buffer de entrada de um processo, mas ele não a recebe.
Arbitrary (*Byzantine*)	Processo ou canal	Processo/canal exibe comportamento arbitrário: ele pode enviar/transmitir mensagens arbitrárias em tempos arbitrários, cometer omissões; um processo pode parar ou seguir um passo incorreto.

Falhas de tempo

Classe de Falha	*Afeta*	*Descrição*
Clock	Processo	Relógio local do processo excede os limites de sua taxa de diferença do tempo real.
Performance	Processo	Processo excede os limites no intervalo de tempo entre dois passos.
Performance	Canal	A transmissão de uma mensagem leva mais tempo do que o limite estabelecido.

Comunicação confiável

- Garantia de entrega – qualquer mensagem no buffer de saída é eventualmente entregue ao buffer de entrada
- Integridade – a mensagem recebida é idêntica à enviada e entregue exatamente uma vez

Como um serviço de comunicação fornece garantia de entrega e integridade?

Através de mecanismos de *check-sum* e *ACKnowledgment*

Modelo de Segurança

- Proteger processos e canais de corrupção
- Proteger recursos (encapsulados por objetos) de acesso não autorizado

Requisitos

- Canais de comunicação devem ser seguros quanto a escuta secreta (*eavesdropping*) e interferência no conteúdo
- Servidores devem poder verificar a identidade de seus clientes
- Clientes devem poder verificar a autenticidade de servidores
- A identidade do originador de uma mensagem deve poder ser verificada após o envio da mensagem - como *assinaturas*

Métodos

- Criptografia para permitir que um par de processos possa estabelecer um canal de comunicação seguro baseado em uma chave
  - Criptografia – ciência de manutenção de segurança de informação
  - Encriptar – embaralhar uma mensagem para esconder seu conteúdo
- Serviço de autenticação para permitir que clientes e servidores possam fornecer uns aos outros evidências convincentes de suas identidades

 19

Ameaças

- Os projetistas de sistemas seguros devem
  - construir uma lista de ameaças (métodos pelos quais políticas de segurança podem ser violadas)
  - mostrar que cada uma das ameaças listadas é tratada pelos mecanismos de segurança aplicados
  - tal demonstração pode ser informal ou, de preferência, formal (lógica)
  - nenhuma lista é exaustiva

 20

Classificação de ameaças

4 grandes classes:

- Vazamento: aquisição de informação por receptores não autorizados
- Interferência: alteração não autorizada de informações (inclui programas)
- Roubo de recursos: uso de facilidades sem autorização
- Vandalismo: interferência na operação de um sistema sem ganho para o intruso

 21

Métodos de ataque (1/2)

- Escuta secreta (*eavesdropping*): obtenção de cópias de mensagens através de leitura (escuta) sem autorização
- Mascaramento: envio ou recepção de mensagem usando a identidade de um objeto sem sua autorização

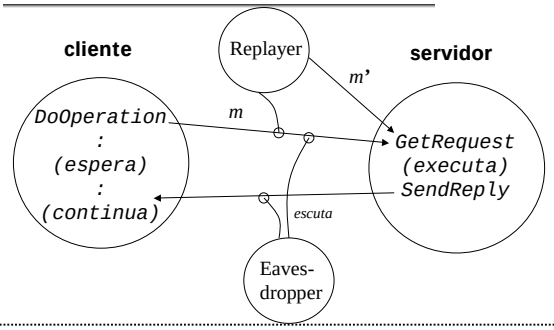
 22

Métodos de ataque (2/2)

- Interferência de mensagem (*message tampering*): interceptação de mensagem e alteração de seu conteúdo antes de passar adiante para o receptor correto
  - difícil de acontecer em ambientes *broadcast* (ex: Ethernet), uma vez que as mensagens são entregues a todas as estações
- *Replaying*: armazenamento de mensagem e envio posterior, quando uma autorização para uso de um recursos é revogada - não mais necessária
  - *replaying* pode ser usado para roubo de recurso ou vandalismo

 23

Exemplo



 24