Amdia de Azevedo Batala
8/10/2023
SD

Vantagens SD
Performance
Availability
Reliability
Dispertion
Scalability

Prop. canais com.
connection based
stream v.s. msg
order
reliability
flux control
Nº recetores
atrasos na rede

Data distribution Service - DSS
- Publisher/subscriber;
- 1 para muitos de forma assincrona;
- Data-centric programming Model
- Eficiente | Standart para API | Menos camadas | Parametros para QoS | scalable, eficiente e previsivel

Modelos cooperação

cliente server
- servidor tem inf. client pede
- Acionado pelo recetor (cliente)
- Unicast (1 para 1)
- Pode ser sincrono ou assincrono
- Naming service
- RPC, RMI, Cobra, ROS

Producer consumer
- Produtor dissemina inf. consumidor usa inf.
- Acionado pelo transm. (produtor)
- Broadcast (para todos nós)
- Assincrono
- CANopen
- comm. outside PC loop

Publisher-subscriber
- Group communication
- Nós aderem a grupos:
  - publisher → Produz inf
  - subscriber → consome inf
- Acionado pelo publisher
  - dissemina info para o grupo
- Assincrono
- RTPS; DDS, ROS, MQTT, OM2M

Shared Memory
- Ler e escrever de um local partilhado; → Area comum é chamada de blackboard e pode estar noutro PC. | comm. dentro do PC loop | Assincrono | adequado para partilha estados
- Real time Data Base : Nós escrevem no mesmo local que é replicado para os agentes (acesso local)

Concurrency
- ↑ Performance | ↑ usability;
- Handle several request simultaneously

Arquitetura	Paral.	I/O op.	Prog.
Iterative	No	Blocking	easy
Mult-thread	Yes	Non-block	race
Event driven	Yes	Non blo	event

Iterative
- 1 thread
- Processa 1 conexão
- Pode bloquear num pedido
- Temporalmente ineficiente

Multiprocessador → Acoplamento forte entre processadores → Partilha de recurso por barramentos paralelos
SD → Partilham canais de comm.

Multi-thread
- cada thread Processa um pedido
- caso threads > processadores é possivel agendar para prevenir bloqueios (1 thread bloqueia outra corre no seu lugar)
- Uma thread para aceitar pedido e varias para processar.
- Necessário assegurar acesso seguro à memória
- Podem existir deadlocks

Event-driven
- waits por evento | Processa evento (sequencial)
- Evita bloqueio usando non-bloquing IO operations
- State Machine Approach (STM)
- Cada processo é encarado como uma Maqui. Estado
- Dividir os processos em potenciais bloqueios
- Não há racing condition

Resource Virtualization
- Allows to emulate the behavior of a system.
- Permite SW desenvolvido para uma arquitetura noutra.
- Process virtual Machine : Permite executar uma única aplicação
- Virtual Machine Monitor: Permite varias aplicações
- Em SD: melhora segurança e confiabilidade e simplifica a gestão

Multicast

IP Multicast
- Para o transmissor é como UDP unicast
- Usa opont broadcast
- O router reencaminha
- O recetor tem que subscrever.

App. level Multicast
- Overlay Network
- spanning tree
- Enviar msg para a raiz
- Passada entre nós por unicast

Epidemic Protocols
"limited flooding" envés de uma "spanning tree"; troca msg com os vizinhos com o fim de alcançar todos os nós. "Lazy way", not immediat. escalavel, tolerante a falhas, lento

Switching trees
- Mudar a topologia da árvore
- um nó poder mudar de pai desde que não esteja na subtree.
- Banyan tree Protocol (BTP)
- Muda por 1 hop
- Quando 1 nó se junta torna-se filho da raiz
- Se um nó falha os seus filhos passam a filhos da raiz
- Para mudar de pai, um nó tem que pedir ao novo

Gossiping
- Um nó tem nova info
- Escolhe outro nó random
- caso o nó já tenha a info. aumenta a prob de parar de propagar
- É possivel que certos nós não tenham a informação.

Anti-entropia
- Um nó escolhe outro para partilhar inf.
- Repete o processo.
- A velha inf. circula, mas dá prioridade à nova.
- Push ⇒ início rápido
- Pull ⇒ Fim rápido
- Push-Pull ⇒ Nós trocam msg.

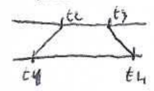
Clock synchronization

Externally
- External sources update
- accurate
- GPS | short term

internally
- Node come with a global clock
- Precision
- Long term

Jitter
Jitter: $\varepsilon = d_{max} - d_{min}$ | $\delta \geq \varepsilon(1 - 1/N)$

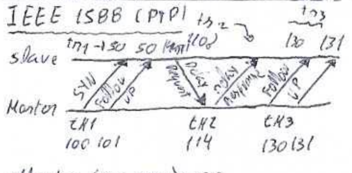
Round-trip Delay (RTD): $((t_4 - t_1) - (t_3 - t_2))/2$; correct delay



$$delay \overset{\delta}{=} \frac{RTD}{2} = \frac{(t_4 - t_1) - (t_3 - t_2)}{2}$$

Offset : $offset = \frac{(t_2 + t_3) - (t_4 + t_1)}{2}$

IEEE 1588 (PTP)



offset = (50 - 100) = 50

delay = $\frac{110 - 114}{2} = -2$

offset + delay

correct offset and delay

Rate correction wit Feedback
$$\rho_m = \frac{1 - ((C_p(s_m) - C_m(s_m)) + d)}{T}$$

ρ_m: Microtick correction term

T : sync interval

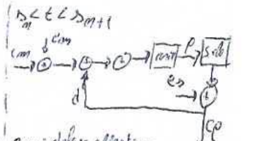
$C_p(t) = C_p(s_m) + \rho_m \times (t - s_m)$

Offset = $D_i(t)$ | Drift $\xi(t, \Delta t)$ | Accuracy = α
Drift rate : $\rho(t)$ | Precision: δ

C_p → local clock

C_m → Master clock

$s_m < t < s_{m+1}$



e_m = delay affecting master
e_s : erro one slav.
d : Network delay

Distributed clk sync
- Node exchange clk values
- virtual clk ref C_m (average)
  - ↳ $C_m = \frac{1}{N}\sum_{k=1}^{N} C_{pk}(s_m)$
- $\delta_{dist} \geq (\xi + \varepsilon)\frac{N}{N-1}$
- crashes can be remove

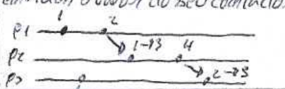
Fault Tolerante Average (FTA)
- Eliminates the most advanced and most delayed clocks from average.
- $\Pi = k\frac{\delta dist}{N - 2K}$ convergence factor
- $\delta dist \geq (\xi + \varepsilon)\frac{N - 2K}{N - 3K}$

Nodes converge to a virtual reference ↑ π ⇒ fast sync ⇒ Good global time

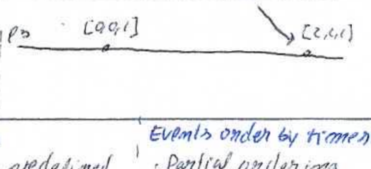
interactive consistency: Nodes send their view of other ~~Nodes send~~ clocks | easy to spot errors | exclude falty measure | tolerant to any Byzantine clocks.

Logical clocks

Lamport clock
- conta eventos e não tempo absoluto;
- cada processo tem o seu contador incrementado a cada evento;
- Quando dois processo comunicam enviam o valor do seu contador;



vector clock
- vetor com visão de todos os processos



Global clock - requirements
- Chronoscopic behavior
  - ↳ No descontinous
- Known Precision
  - ↳ Maximum intervals between ticks of 2 Nodes
- High dependability

Limits of time measure
- SD com sistema baseado em granularidade
- ~~Aparece~~ Eventos no mesmo ts caso $\Delta t < g$
- $d_{obs} - 2g < d_{true} < d_{obs} + 2g$ (d_{obs} (observed duration))
- $d_{obs} > 2g$ or $\{e\}$ is 3g

Dense time base
- Any events can occur at any time.
- The obs. of an event by 2 nodes require a consensus protocol
- Migh be impossibel to recover temporal order

Sparse time base
- Events occur in predefined intervals
- Events controled by a computing system
- Can't control certain events (ex. faults)
- Time-trigger v.s. Event trigg.

Events order by timestamps
- Partial ordering
- global time has granularity g
- it is sparse with g = 3g
- Events restricted to sparse time
- Events outside system are maped based on consensus protocol
- Can achive this wit logial clocks.

Cryptography → Confidentiality; Integrity; Availability

DES (shared key)

• 2 basic operations
↳ Permutation
↳ substitution

• 16 rounds; diferent key of 48 bit generate from 56-bit masterkey

• Work by mangler fuction
↳ Expands R to 48 bit
↳ XOR with round key
↳ divides into 8 blocks of 6 bits each
↳ substitution: 6→4 bits
↳ 8 blocks of 4 bits are combined into 32 bit block

• used in encryption and decryption

RSA (Public key)

• p e q são primos

• $m = p \times q$; $z = (p-1)(q-1)$

• $de = 1 \bmod z$

• $x^{d.e} = x \bmod m$; x é a palavra

• encriptação
↳ $c = b^e \bmod m$; b é palavra

- desencriptação
↳ $b = c^d \bmod m$

• key
↳ $K_e = (e, m)$ → public
↳ $K_d = (d, m)$ → private

• Difícil de fatorizar nº longos (m)

- chaves de 2048 bits

Cryptographic Hash Fuction

• Usados para Integridade

• Hash function h() deve:
↳ compressão: Input com tamanho variável mapeado para valor hash de tamanho fixo.
↳ Easy of computation
↳ Não reversível
↳ Resistente a colisões: valor x tem sempre o mesmo hash; dois valores diferentes têm hash diferente

• K stages (K é de 512 bits)
↳ entra nº 128 bits e bloco de 512 bits
↳ sai 128 bit
↳ tem 4 rondas

Msg Authetication com hash

• Adicionar um "key" à msg
↳ hash value da msg
↳ msg authentication code (MAC)

• Key is shared by all.

Digital signature

- Indentify the author
• be verifiable by others

• On point-to-point MAC allows authen. but does not allow others.

- MAC does not provide a prevention from the author to reject.

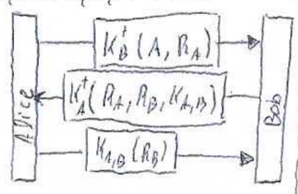
- Digital signatures are based on asymetric encryption.

Authetication

Public key Authetication

• Alice e Bob sabem as sua public key

• Apenas o propriatário sabe a private key



} R_A e R_B são desafios; $K_{A,B}$ é uma key session

- Não garante integridade

• O atacante pode alterar o conteúdo das msg

• Usando MAC asseguramos integridade

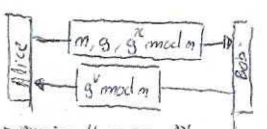
• session key garante autenticidade
↳ Usar a mesma chave ⇒ replay attack
↳ Usar a mesma chave → + info attacker
↳ Se encriptar, caso atacante descobra a chave, não consegue desincriptar.

Diffie-Hellman Key-Agreement

• n large prime number

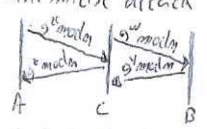
• g < n; both public

• each one choses a private large prime nº: x and y



• session key is $g^{xy} \bmod m$

• vulnerable to man-in-the midle attack



- Publish $g^x \bmod m$ } prevent
- Authenticate } MIM

Public Key Certificate

• Problem associating public key to user

• if C convices A that B's public key math C private it can impersonate B

- certificates contain
↳ name (principal's)
↳ Public key
↳ signature (by CA)
↳ name of CA

• Have expire dates

Reliability probability a system has not to fail until time t (MTTF) mean time to failure

Availability probability that a system is working correctly (MTTR) mean time to repair $a = \frac{MTTF}{MTTF + MTTR}$

Failure Models

• Crash: stops responding

• Omission: don't respond to some inputs (ex. lose msg)

• Timming/Performance: does not respond on time

• Byzantine/arbitary: behave in an arbitrary way

Atomic commitment

AC1 - All process that decide, decide the same value

AC2 - The decision is final

AC3 - If a process commits them all voted commit

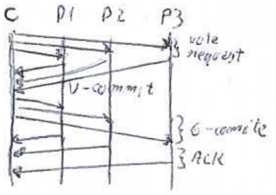
AC4 - if all voted commit the all go for commit

AC5 - if failure that system can tolerate ocur, them all process reach a decision

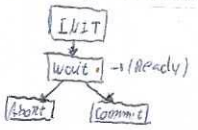
Two-Phase commit

• Only one coordinator and several participants.

• coordinator can be parti.



• Se 1 aborta abortam todos



Timeouts

• Coord. wait in "wait" and abort

• Participant:
↳ "INIT" decides abort
↳ "Ready" termination

Termination

- Nós falam para decidir

• caso 1 vote aborte, abortam todo

• Caso 1 tenha Global, seguem esse comando

• Caso contrário, espera por resposta

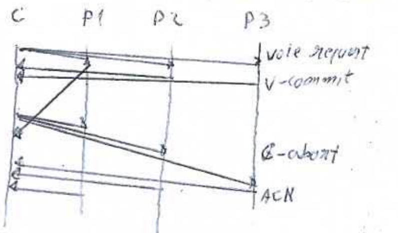
Recovery

• Quando um nó recupera tem que terminação

• Caso crash enquanto aguarda pela msg.
↳ timeout
↳ termination

• Caso contrário → Abort (coord. em INIT)

• Para que um nó recupere tem que guardar estado



Election - Assumptions

• Todos os nós operam e usam o mesmo algoritmo

AC4 - Todos os nós têm o mesmo storage.

AC5 Quando um nó falha para os processos todos

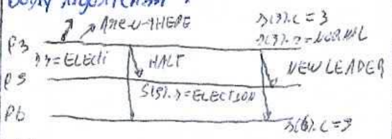
AC6 Não há erros na comun. mas há perdas de msg

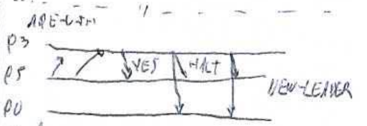
AC7 msg entregues em ordem

AC8 A comm. não falha tem um tempo limite T para entregar msg

AC9 Um nó responde sempre

Bully Algorithm *

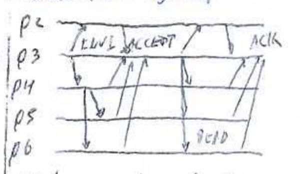




Failures

Inicia Eleição quando
↳ falha no líder ou candidato
↳ Recuperação do nó

Invitation algorith



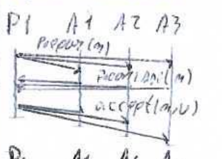
- Nodes 3 and send INV to other nodes INVITATION (3,9)

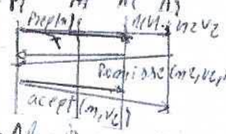
• Groups: 4,5,6; 3,2

• P4 fowards INV

• Node 1 crash | {2,3} group | Node 1 wak and form group with node 2 | Node 2 can't be in 2 groups

Synod's Execution





• Always choses the higher proposal value

- Majority decides

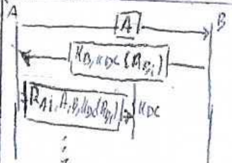
Paxos: State machine replication

• Eleger líder | fase 1 para todos

• Respont Promisse

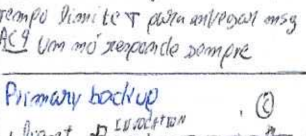
• Líder recebe pedido cliente

• começa fase 2



{Needham - Schroeder's}

Primary backup



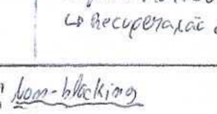
Primary fails

• (C) → transparent to client

• (A) → No backup → No update
→ New request ⇒ New primary

• (B) → Some backup migh update

Non-blocking



• Menos costant

View sync

• Recebe Pedido | Executa
• Emite update

• Recebe suficientes ACK responde

• Quando falha elege

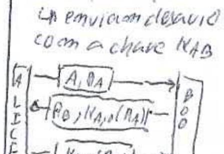
- Detect fails by "I'am alive" msg.

* Bully → Node looks for stronger leader | imposes it self → brags about it.

If we chop AC8 and AC9 the Assertions 1 and 2 can not be satisfied | Assume node i is coordenator does not respond
↳ From others node view, i crashe ↳ Elect new leade ⇒ 2 coordinator ⇒ violate assertion 1

Secret key Authlication

• Partilham chave
↳ enviam desafio encriptado com a chave K_{AB}



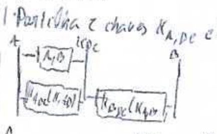
• Vulnerável a Reflection
↳ repetir com a chave do outro

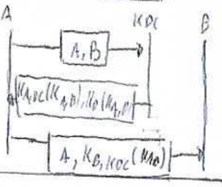
KDC

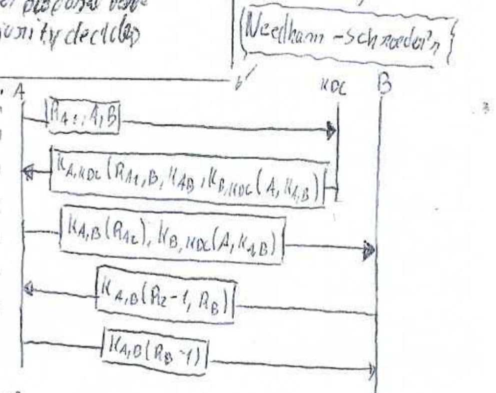
• Não é scalable

- Mediador

• Partilha 2 chaves $K_{A,DC}$ e $K_{B,DC}$






A
KDC
B
R_{A1}, A, B
$K_{A,KDC}(R_{A1}, B, K_{AB}, K_{B,KDC}(A, K_{A,B}))$
$K_{A,B}(R_{A2}), K_{B,KDC}(A, K_{A,B})$
$K_{A,B}(R_2 - 1, R_B)$
$K_{A,B}(R_B - 1)$

• R_A is used to make sure A talks to KDC (prevent replay att.

• KDC implements B in response to prevent impersonat.

• Vulnerability: C learn key: $K_{A,DC}$ → C impersonate A