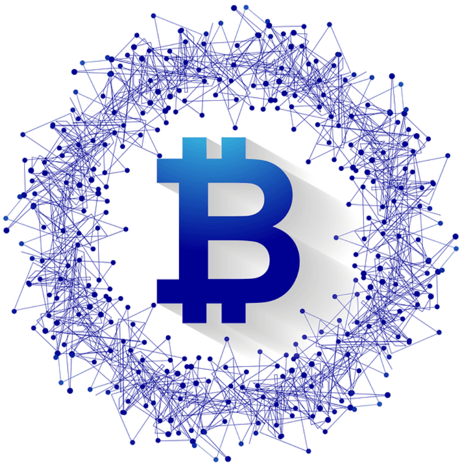


Fundamentos

Generais Bizantinos

Prof. Dr. Bruno de Carvalho Albertini

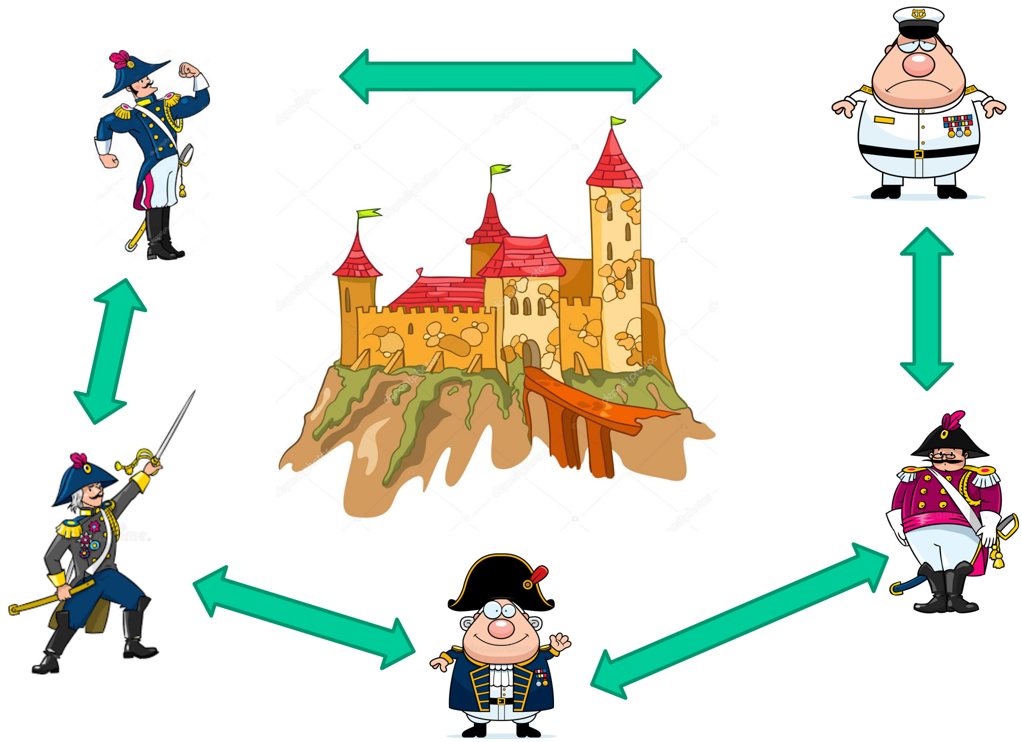
Departamento de Engenharia de Computação e Sistemas Digitais Escola Politécnica da USP

Motivação

- "Um sistema computacional deve ser capaz de lidar com a falha de um ou mais dos seus componentes"
- Um sistema computacional falho em BC
  - Envia mensagens conflitantes para diferentes partes do sistema
  - Não envia algumas mensagens

Generais Bizantinos

Generais Bizantinos



Generais Bizantinos

- Algumas divisões do exército bizantino estão sitiando uma cidade, cada uma com seu general
- Os generais podem se comunicar uns com os outros trocando mensagens
- Somente um ataque de 50%+1 ganha
- Alguns generais podem ser traidores!
- Pode-se assumir que há um general comandante, e os demais (que forem leais) acatam a ordem do comandante

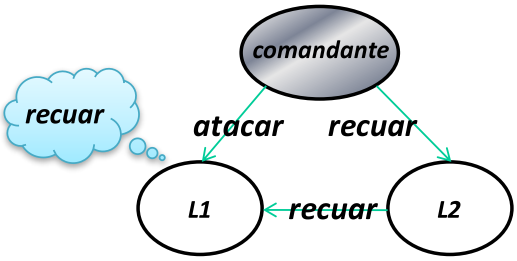
Objetivo

- Todos os generais leais decidem o mesmo plano de ação
- Um grupo pequeno (<50%+1) não deve influenciar os generais leais a tomar uma decisão ruim

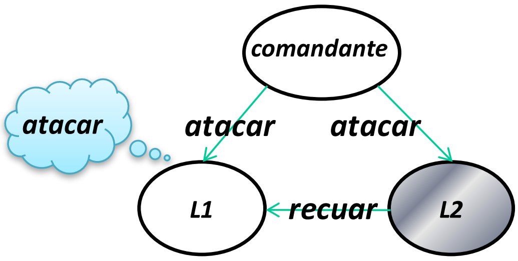
Formalmente: BGP (Byzantine Generals Problem)
1.Todos generais leais obedecem a mesma ordem
2.Se o comandante é leal, todos os generais leais obedecem sua ordem

Número máximo de traidores

Adotaremos que os generais devem decidir um único bit:
1 para atacar e 0 para recuar (recuar é o padrão na dúvida).



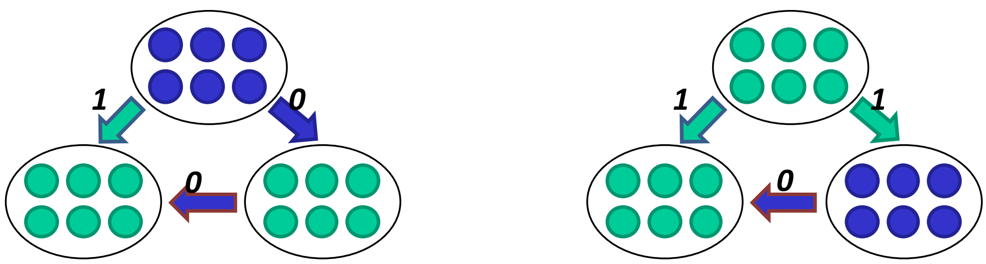
Caso 1:
Comandante é traidor



Caso 2:
General L2 é traidor

Número máximo de traidores

- **Corolário 1:** No caso de 3 generais, não há como lidar com um traidor
- **Corolário 2:** Não há solução quando menos que 3m+1 generais precisam lidar com m traidores



Solução com mensagens

BGP: Byzantine Generals Problem

1. *Todos generais leais obedecem a mesma ordem*
2. *Se o comandante é leal, todos generais leais obedecem sua ordem*

- Não há solução com menos que 3m+1 generais quando precisam lidar com m traidores
- Há uma solução quando o número de generais leiais é > 3m (paper do Leslie Lamport)

Premissas:

1. ***Todas as mensagens enviadas chegam corretamente***
2. ***O receptor sabe quem enviou***
3. ***A falta de uma mensagem pode ser inferida***

Algoritmo solução com mensagens

Definição recursiva, caso base m=0, recursivo para m>0:

Algoritmo OM(0)

1. O comandante envia a sua ordem para cada general
2. Cada general usa o valor recebido do comandante

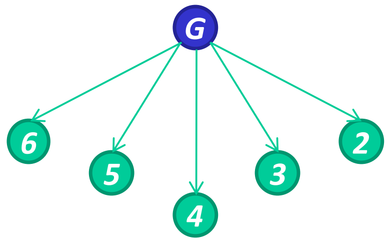
Algoritmo OM(m), m > 0

1. O comandante envia a sua ordem para cada general
2. Para cada i, seja v_i a ordem que o general i recebeu do comandante. O general i agirá como comandante recursivamente em OM(m-1) para enviar o valor v_i para cada um dos n-2 generais restantes
3. Para cada i, e para cada j ≠ i, seja v_j a ordem que o general i recebeu do comandante j no passo 2 (usando OM(m-1)). O general i usa o valor *Majority*(v_1, v_2, … v_n).

Exemplo (1)

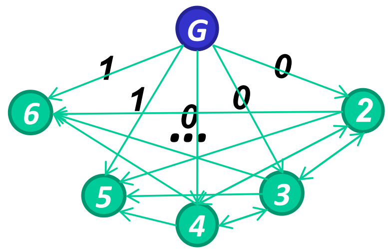
m=0

O comandante envia a mensagem para todos generais.



m=1

Cada general envia a mensagem que ele recebeu para todos os outros generais



Sender=P_2		Sender=P_3		Sender=P_4		Sender=P_5		Sender=P_6	
Dest	Msg	Dest	Msg	Dest	Msg	Dest	Msg	Dest	Msg
P_2	{0,12}	P_2	{0,13}	P_2	{0,14}	P_2	{1,15}	P_2	{1,16}
P_3	{0,12}	P_3	{0,13}	P_3	{0,14}	P_3	{1,15}	P_3	{1,16}
P_4	{0,12}	P_4	{0,13}	P_4	{0,14}	P_4	{1,15}	P_4	{1,16}
P_5	{0,12}	P_5	{0,13}	P_5	{0,14}	P_5	{1,15}	P_5	{1,16}
P_6	{0,12}	P_6	{0,13}	P_6	{0,14}	P_6	{1,15}	P_6	{1,16}

Sender=P_2	Sender=P_3	Sender=P_4	Sender=P_5	Sender=P_6
{0,12}	{0,13}	{0,14}	{1,15}	{1,16}

Exemplo (2)

Mensagens do passo 1:

Sender=P_2	Sender=P_3	Sender=P_4	Sender=P_5	Sender=P_6
{0,12}	{0,13}	{0,14}	{1,15}	{1,16}

Passo 2: Cada general envia a mensagem que ele recebeu para todos os outros

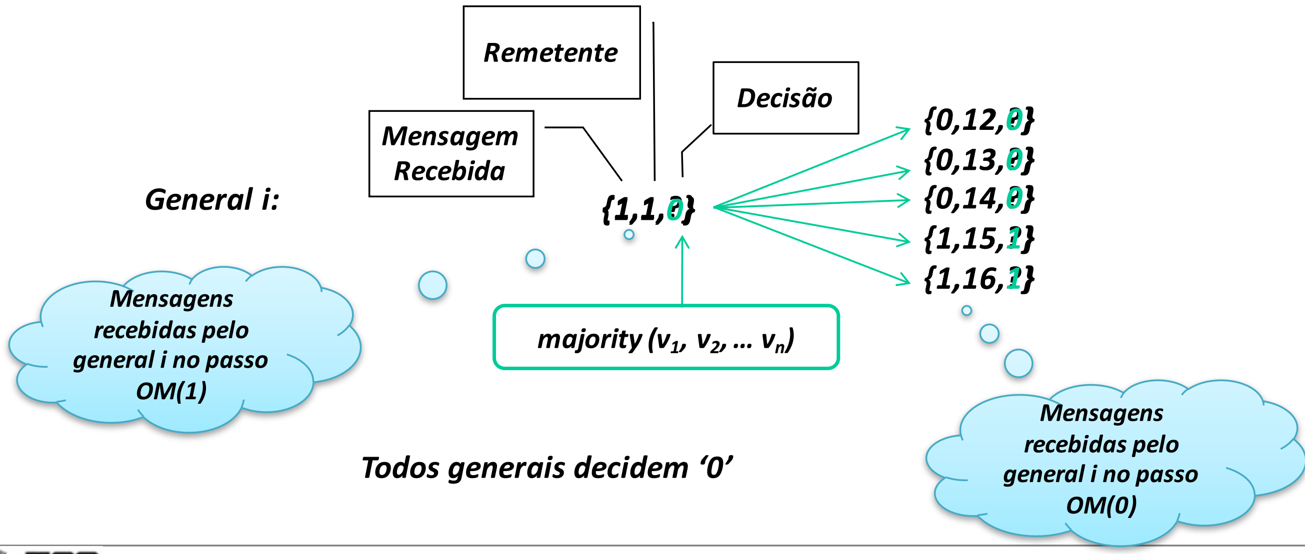
Sender=P_2	Sender=P_3	Sender=P_4	Sender=P_5	Sender=P_6
{0,132}	{0,123}	{0,124}	{0,125}	{0,126}
{0,142}	{0,143}	{0,134}	{0,135}	{0,136}
{1,152}	{1,153}	{1,154}	{0,145}	{0,146}
{1,162}	{1,163}	{1,164}	{1,165}	{1,156}

Estas são todas as mensagens enviadas no algoritmo recursivo

Qual será o resultado?

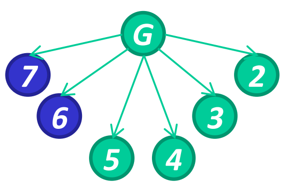
Exemplo (3)

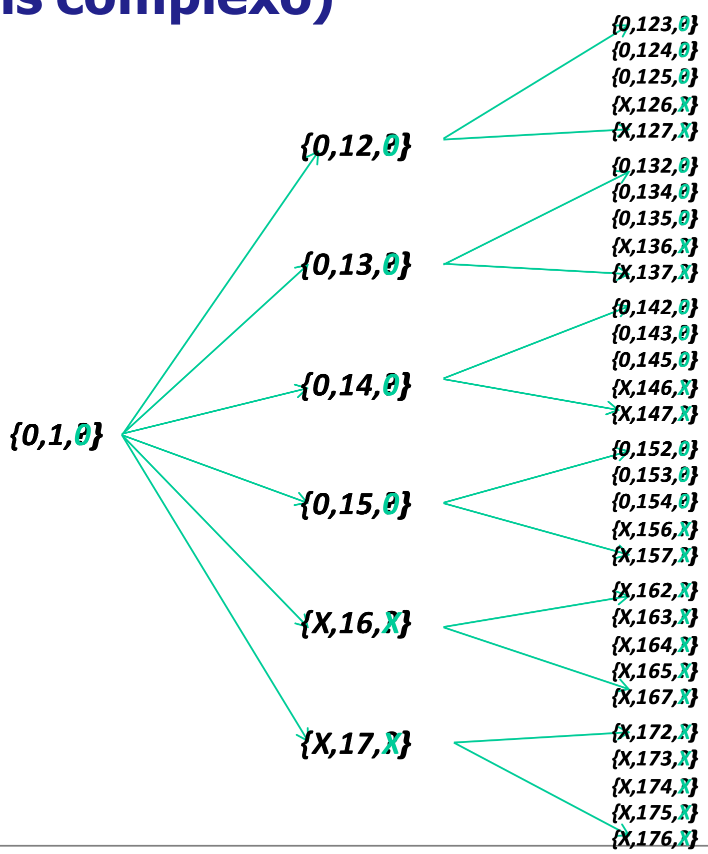
- Pode-se construir uma árvore de recursão para a decisão de cada general:



Exemplo (mais complexo)

- n=7 , m=2





Oral Messages Algorithm: Proof (1)

The Byzantine Generals Problem

1. ***All loyal lieutenants obey the same order***
2. ***If the commander is loyal, then every loyal lieutenant obeys the order he sends***

Lemma 1:

For any m and k, Algorithm OM(m) satisfies (2) if there are more than 2k+m generals and at most k traitors

Proof: (by induction on m)

Base: Algorithm OM(0) satisfies (2) when the commander is loyal.

Assumption: the algorithm OM(m-1) satisfies (2) if there are more than 2k+m-1 generals and at most k traitors

Step:

- In step (1) every loyal commander sends the value 'v' to all n-1 lieutenants.
- In step (2) each loyal lieutenant applies OM(m-1) with n-1 lieutenants
- By hypothesis, $n > 2k + m \Rightarrow n - 1 > 2k + (m-1) \geq 2k$
- ⇒ A majority of the n-1 lieutenants are loyal
- ⇒ By assumption, each loyal lieutenant has v_i = 'v' for a majority of n-1 values i.
- ⇒ Majority(v_1,…v_n)='v' in step (3).

Oral Messages Algorithm: Proof (2)

The Byzantine Generals Problem

1. ***All loyal lieutenants obey the same order***
2. ***If the commander is loyal, then every loyal lieutenant obeys the order he sends***

Theorem 1:

For any m, algorithm OM(m) satisfies conditions 1 and 2 if there are more than 3m generals, and at most m traitors.

Proof: (By induction on m)

Base: if there are no traitors, OM(0) satisfies conditions 1 and 2

Assumption: OM(m-1) satisfies conditions 1 and 2 if there are more than 3(m-1) generals, and at most m-1 traitors

Step:

- We can use lemma1 with k=m, and get that condition 2 holds.
- Condition 1 follows from condition 2 when the commander is loyal.
- Else, there are at most m traitors and the commander is one of them
- ⇒ At most m-1 of the lieutenants are traitors
- ⇒ At step (2) of the algorithm there are 3m-1 > 3(m-1) generals, and at most m-1 traitors
- ⇒ From the assumption, OM(m-1) satisfies conditions 1 and 2.
- ⇒ All loyal generals get the same values v_j for every loyal general j.
- ⇒ Majority(v_1,…v_n) is the same for all loyal lieutenants in step (3).

QED

Será que não há algo melhor?

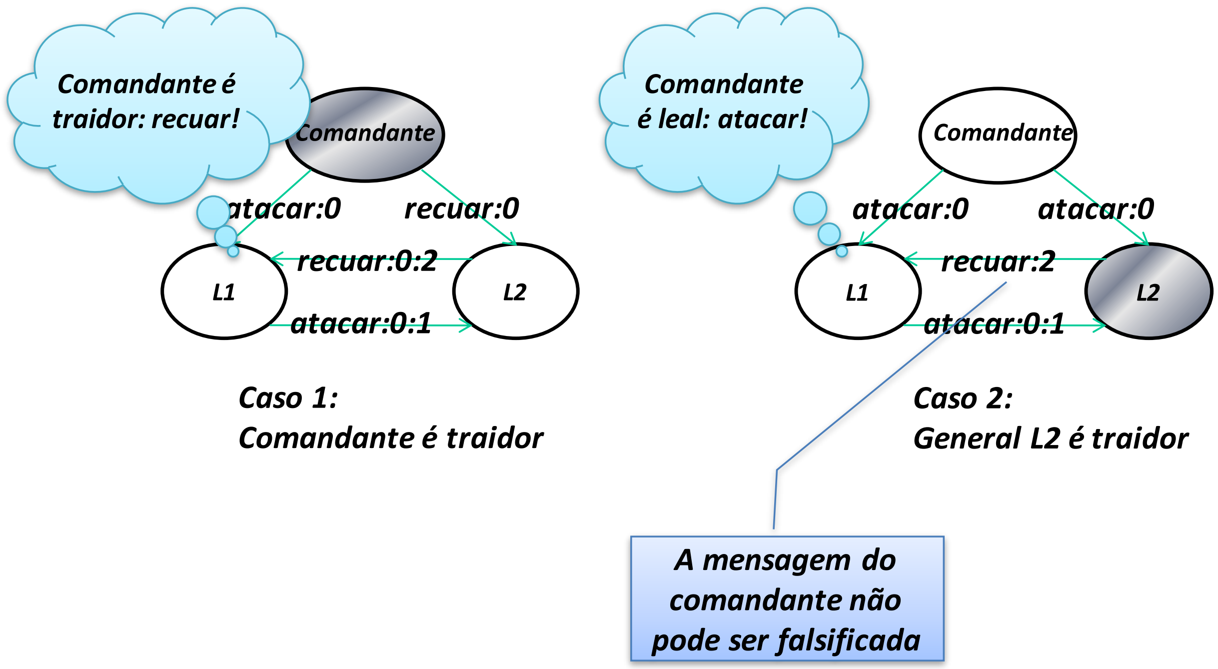
Solução com assinaturas

- Problema (abstrato): traidores mentem

Premissa (nova):
- ***A assinatura de um general leal não pode ser forjada e qualquer alteração em uma mensagem assinada será detectada***
- ***Qualquer um pode verificar a assinatura em uma mensagem***

O limite mínimo passa a ser ($n \geq m+2$)

Exemplo



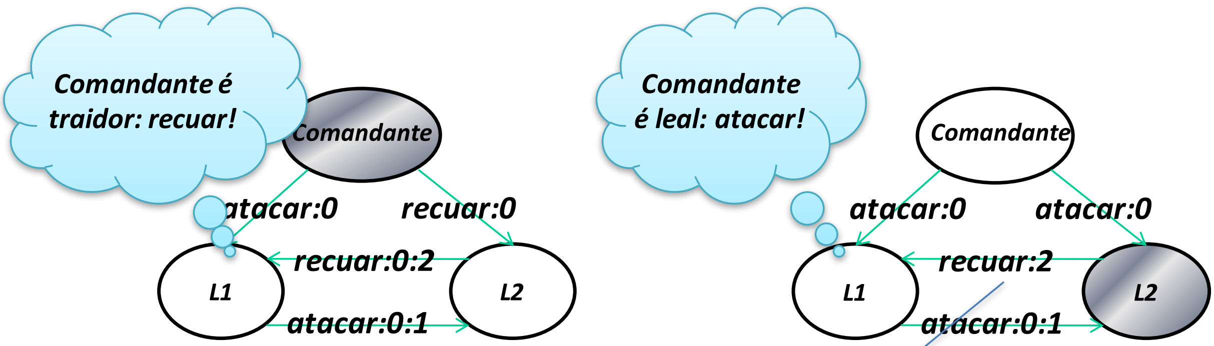
Algoritmo Assinado

Algoritmo SM(m)

Inicialmente V_i = {}

1. O comandante manda sua ordem para cada general
2. Para cada i:
    1. Se o general i recebe uma mensagem na forma v:0 do comandante e não recebeu nenhuma ordem ainda, então
        1. $V_i \leftarrow \{v\}$
        2. Envia mensagem v:0:i para todos os outros generais
    2. Se o general i recebe uma mensagem na forma $v:0:j_1: \ldots :j_k$ e v ainda não esntá no conjunto V_i então
        1. $V_i \leftarrow V_i \cup \{v\}$
        2. Se k<m então envia a mensagem $v:0:j_1: \ldots : j_k:i$ para todos os outros generais exceto $j_1,\ldots,j_k$
3. Para cada i: quando o general i não receber mais nenhuma mensagem (já recebeu de todos), executa a ordem *choice*(V_i)

Exemplo



Caso 1:
Comandante é traidor

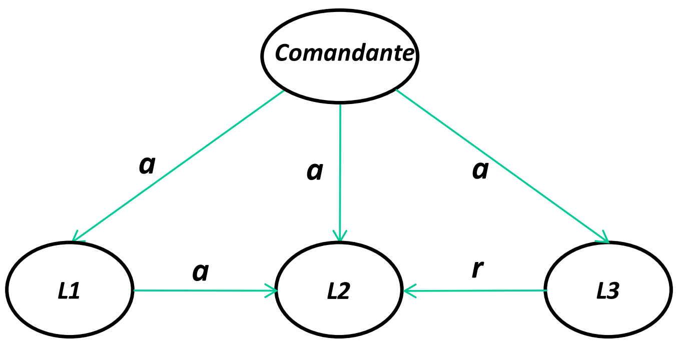
V(L1) = {atacar, recuar}

Caso 2:
General L2 é traidor

V(L1) = {atacar}

A mensagem do comandante não pode ser falsificada

Exercício



Conclusões

- As soluções para o BGP são caras (3M)
- Usa redundância nas mensagens para obter o consenso
- Problemas se >1/3 dos nós estiverem comprometidos

Dúvidas? Tem uma solução melhor?

Signed Messages Algorithm: Proof

The Byzantine Generals Problem

1. ***All loyal lieutenants obey the same order***
2. ***If the commander is loyal, then every loyal lieutenant obeys the order he sends***

- If the commander is loyal, then he sends his signed order v:0 to every lieutenant in step (1), and every loyal lieutenant will add v to V_i.
- Since no traitorous lieutenant can forge a message of the form v':0, a loyal lieutenant can receive no other order in step (2.2).
- For all loyal lieutenants: V_i = {v}

  => every loyal lieutenant obeys the order the general sends. (condition 2 OK)
- It remains to prove condition 1 for the case where the commander is not loyal.
- Two loyal lieutenants i and x obey the same order in step (3) if the sets V_i = V_x .
- $v1 \in V_i$ => i received the message v1:
  - If it was received from the general – It was sent to x in step 1. $v1 \in V_x$
  - It was received by v1:0:{list}. If x is in the list, then x has .
  - It was received by v1:0:{list}, and x is not in the list:
    - If one of the lieutenants in the list is loyal, then x received it when the loyal lieutenant sent it
    - There are at most m-1 traitorous lieutenants, so in step m lieutenant i will send the message to x.

QED