Problema de los generales bizantinos

El **problema de los generales bizantinos** es un experimento mental para plantear, de una forma metafórica, el problema que se da entre un conjunto de sistemas informáticos que tienen un objetivo común. Deben encontrar un plan de acción común a partir de una estructura jerárquica, donde uno de los sistemas que tiene mayor rango proporciona una orden a partir de la cual el resto de sistemas tiene que operar (fijar su decisión). Además es posible que alguno de ellos no sea fiable y provea información falsa de forma intencionada.[1] [2]

Índice

Planteamiento del problema

Supongamos un escenario de guerra en el que tenemos un grupo de **m** generales bizantinos que están asediando una ciudad desde distintos lugares y tienen que ponerse de acuerdo para atacar o retirarse de forma coordinada. Entre los generales hay solo uno que puede cursar la orden por ser el **comandante**. El resto se dice que son **tenientes**.

Los tenientes se comunican entre ellos cuando reciben la orden del comandante y las dos posibles órdenes del comandante son "atacar" y "retirarse".

Uno o más de los generales puede ser un **traidor** (al resto se les llama **leales**), por lo que su objetivo es conseguir que todos los generales leales no se pongan de acuerdo. Para ello pueden ofrecer información errónea. Por ejemplo, si el comandante es el traidor, podría mandar órdenes contradictorias a los distintos tenientes. Si el teniente es un traidor podría indicarles a otros tenientes, con el fin de confundirlos y que creyeran que el traidor es el comandante, que el comandante les envió la orden contraria a la que realmente les envió.

Para resolver el problema tenemos que buscar algoritmos que nos permitan conseguir alguno de los siguiente objetivos:[3]

- Todos los tenientes leales toman la misma decisión.
- Si el comandante es leal, entonces todos los tenientes leales realizan la orden que él decidió.

Normalmente para llegar a una solución se suelen hacer las siguientes condiciones adicionales:[3] [2]

- Cada mensaje que se envía llega correctamente.
- Cada receptor de un mensaje conoce quién lo envía.
- La ausencia de mensaje puede ser detectada.
- Ante la ausencia de mensaje se tiene una orden por defecto. Esta condición es para evitar el problema de que el comandante sea un traidor y no envíe órdenes.

Algoritmos solución

En 1982 Leslie Lamport, Robert Shostak y Marshall Pease[4] proporcionaron distintos algoritmos de solución en función de condiciones adicionales.

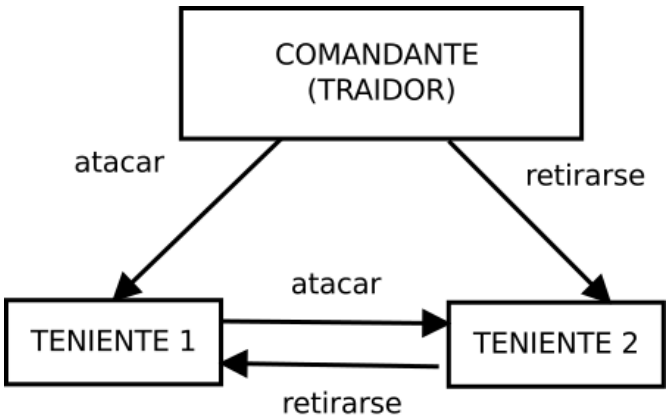
Mensajes orales y todos se pueden comunicar con todos

La estrategia se basa, con el fin de detectar si el comandante es el traidor, en que los tenientes se reenvíen entre sí la información que el comandante les ha mandado. Si el teniente es leal la información que transmitirá el teniente será la que le envió el comandante. La consecuencia de usar mensajes orales (no firmados) es que un general traidor puede decir que el comandante le ha mandado cierta información cuando no es así.

Caso de 3 generales

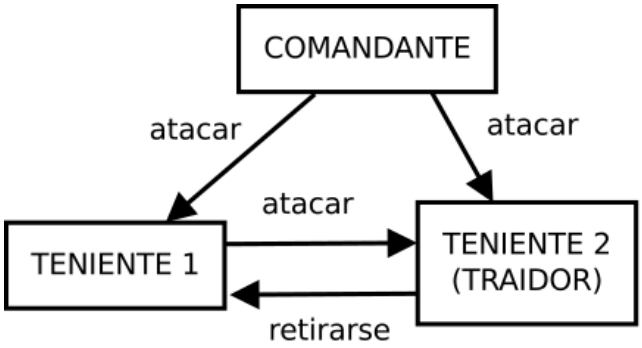
Analicemos el caso en el que tenemos tres generales (m=3).

- Supongamos que el comandante es un traidor. Si el comandante envía una orden distinta a cada teniente entonces habrá un teniente que no sepa qué acción realizar:



Problema de los 3 generales bizantinos con comandante traidor

- Supongamos que un teniente es el traidor. Entonces este retransmite al otro teniente información distinta a la que recibió del comandante. Por tanto el otro teniente no sabrá qué acción realizar:



DILEMA TENIENTE 1 ¿QUIÉN ES EL TRAIDOR?

Problema de los 3 generales bizantinos con teniente traidor

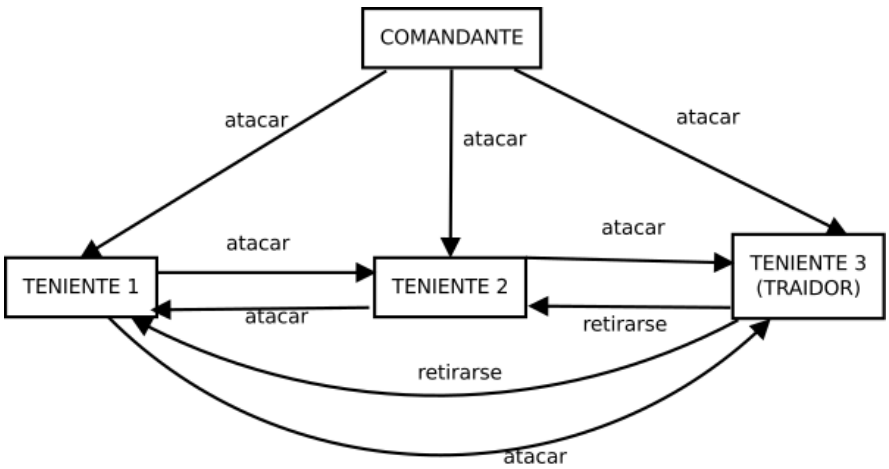
La conclusión es que no existe solución que garantice que se cumplan las condiciones del problema si se permite que con tres generales uno sea un traidor. Esto es debido a que no hay suficientes generales para formar una opinión consensuada.

Caso de 4 generales

Si tuviéramos 4 generales (m=4) sí sería posible el acuerdo a través del siguiente algoritmo:

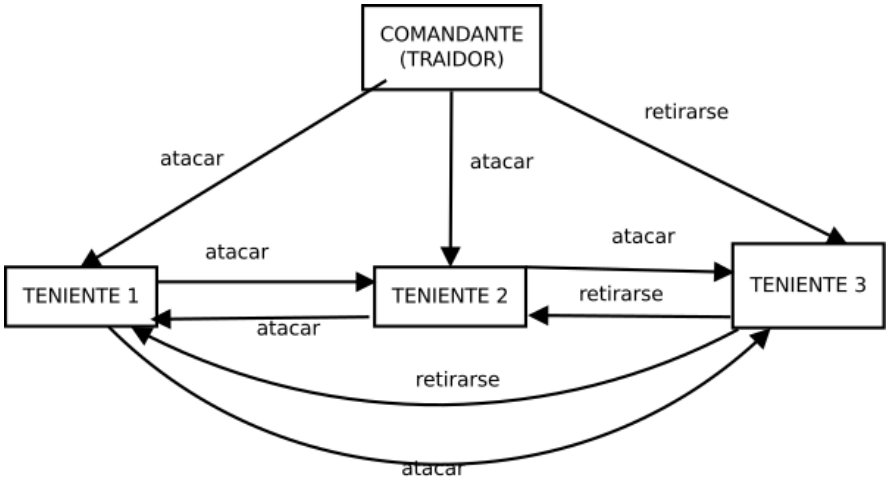
> Al recibir la orden del comandante y los mensajes de los otros 2 tenientes, los tenientes leales decidirán la orden de consenso según la siguiente función de mayoría:
>
> > $M(v_1, v_2, v_3)$: Devolver el valor de v que sea mayoría entre v_1, v_2, v_3.
> > Donde el valor de v_i es la orden mandada desde los distintos generales al general al que estamos evaluando su decisión.

Veamos el esquema si el comandante es leal y un teniente es traidor:



Problema de los 4 generales bizantinos con teniente traidor

Veamos el esquema si el comandante es traidor y los tenientes leales:



Problema de los 4 generales bizantinos con comandante traidor

Caso de m generales

Generalizando a m generales se puede decir que si tenemos t traidores necesitamos que m sea al menos 3t+1. Al algoritmo generalizado se le llama OM(m) (donde las siglas OM vienen del inglés *Oral Messages*) y viene descrito por usar la siguiente función de mayoría:

> $M(v_1, v_2, \ldots, v_n)$: Devolver el valor de v que sea mayoría entre $v_1, v_2, \ldots, v_n$
> Donde el valor de v_i es la orden mandada desde los distintos generales al general al que estamos evaluando su decisión.

Mensajes firmados y todos se pueden comunicar con todos

En este escenario los mensajes van firmados (se trata de mensajes escritos). Al ir firmados no son modificables y por tanto los traidores no pueden modificarlos y decir que provienen del comandante. En esta situación es posible resolver el problema con sólo tres generales y uno de ellos traidor. El algoritmo de este tipo de problemas se llama SM(m) (donde SM viene del inglés *Signed Messages*) y es el siguiente:

> Primero el comandante envía una orden firmada a todos los tenientes. Cada vez que un teniente recibe un mensaje firmado lo guarda, hace una copia, la firma y la reenvía a todos los tenientes que no venían en la firma del documento. Según este algoritmo los generales no recibirán más mensajes cuando tengan todas las posibles combinaciones. Una vez recibidas, cada nodo toma la decisión basándose en la orden transmitida por la mayoría.

En este escenario los comandantes traidores son descubiertos inmediatamente ya que han firmado órdenes contradictorias.

No todos se pueden comunicar con todos

Si falta alguno de los caminos de comunicación las cosas se complican. Veamos los requerimientos tanto cuando hay mensajes orales como mensajes firmados.

- Con mensaje orales el algoritmo OM(m) solo funciona bajo la condición de que el grafo de caminos posibles entre generales sea 3m-regular (cada nodo tiene al menos 3m vecinos), lo que implica 3m+1 nodos (generales). A esta versión del algoritmo se le llama OM(m,p), donde p es el número de vecinos.
- Para mensajes firmados la única condición es que todos los generales leales estén conectados, para que así los traidores no puedan bloquearle y evitar que le pasen o pase la

orden firmada.

Consideraciones

- El problema de los dos generales bizantinos es el mismo que se tiene cuando se realiza una transmisión de dinero sin un intermediario confiable.[1] Bitcoin ofreció la primera solución práctica a este problema.
- En el mundo real las líneas fallan de forma no deliberada. Para detectarlas se pueden usar códigos de detección de errores. En un escenario con mensajes orales, una línea que falla puede considerarse como un nodo traidor. Si se utilizan mensajes firmados entonces un fallo en una línea se detectaría de forma irrefutable.
- Para reconocer al emisor de un mensaje empleando mensajes orales, se deberían tener líneas fijas y no redes de comunicaciones. Con mensajes firmados no hay problema para reconocer al emisor.
- La ausencia de mensajes se suele detectar usando time-out (límites de tiempo).
- En el mundo real nunca está garantizado que un error aleatorio no pueda falsear una firma. Sin embargo esto tiene una probabilidad muy baja con métodos de firma adecuados.
- Evitar fraudes deliberados se convierte en un problema criptográfico. Por tanto es importante elegir algoritmos de firma seguros.
- Se debe detectar si un mensaje se envía dos veces, mediante la comprobación de su firma. De tal modo que una firma no puede ser generada si el proceso ya ha visto esa misma firma en otro instante.

Referencias

1. Bitcoins y el problema de los generales bizantinos (http://web.ua.es/en/recsi2014/documentos/papers/bitcoins-y-el-problema-de-los-generales-bizantinos.pdf). Cristina Pérez-Solà y Jordi Herrera-Joancomart. Universitat Autònoma de Barcelona. 2014
2. Tarea 2. Sistemas Distribuidos (http://documents.mx/documents/tarea-2-sistemas-distribuidos.html) Archivado (https://web.archive.org/web/20170616230829/http://documents.mx/documents/tarea-2-sistemas-distribuidos.html) el 16 de junio de 2017 en Wayback Machine.. Marcelo Valdivia Lagos. 04/04/2013
3. Lamport, Leslie; Shostak, Robert; Pease, Marshall; Lütolf, Manuela. «The Byzantine Generals Problem» (https://web.archive.org/web/20161003122301/http://informatik.unibas.ch/fileadmin/Lectures/HS2012/CS341/workshops/reportsAndSlides/luetolf_report.pdf) (pdf). *Universidad de Basilea* (en inglés). Archivado desde el original (http://informatik.unibas.ch/fileadmin/Lectures/HS2012/CS341/workshops/reportsAndSlides/luetolf_report.pdf) el 3 de octubre de 2016. Consultado el 19 de noviembre de 2019. «This paper discusses what happens if computer systems must find a common plan of action,and it is possible that some of them are faulty and provide bad input. It uses the metaphor ofByzantine armies around an enemy city, who must together decide if they should "attack" or"retreat", while being able to communicate only by messenger, and not knowing if some ofthem might be traitors».
4. The Byzantine Generals Problem (http://research.microsoft.com/en-us/um/people/lamport/pubs/byz.pdf). Leslie Lamport, Robert Shostak y Marshall Pease.SRI International 1982