

Public-Key-Authentisierung mit Byzantine Agreement

Dominik Schürmann

Seminar WiSe 2009/2010
Institut für Betriebssysteme und Rechnerverbund
Abteilung Verteilte und Ubiquitäre Systeme

10. Februar 2010



TECHNISCHE UNIVERSITÄT
CAROLO-WILHELMINA
ZU BRAUNSCHWEIG

Inhalt der Seminaarausarbeitung

Thema: **Security Issues and Countermeasures in MANETs and P2P Networks**

Angriffe und Gegenmaßnahmen in dezentralen Netzen bezüglich...

- Verfügbarkeit
- Integrität und Authentizität
- Anonymität

Kryptografische Grundlagen und Algorithmen:

- Public-Key-Kryptosysteme
- Threshold Signature Schema
- **Authentisierung mit Byzantine Agreement**

Inhalt des Vortrags

- Voraussetzungen
 - Byzantine Generals Problem
 - Public-Key-Signaturschema
- Authentisierungsprotokoll mit Byzantine Agreement

Byzantine Generals Problem

Byzantine Generals Problem

„We imagine that several divisions of the Byzantine army are camped outside an enemy city, each division commanded by its own general. The generals can communicate with one another only by messenger. After observing the enemy, they must decide upon a common plan of action. However, some of the generals may be traitors, trying to prevent the loyal generals from reaching agreement. The generals must have an algorithm to guarantee that

- A. All loyal generals decide upon the same plan of action. [...]*
- B. A small number of traitors cannot cause the loyal generals to adopt a bad plan.“[1, S. 382-383]*

Byzantine Agreement

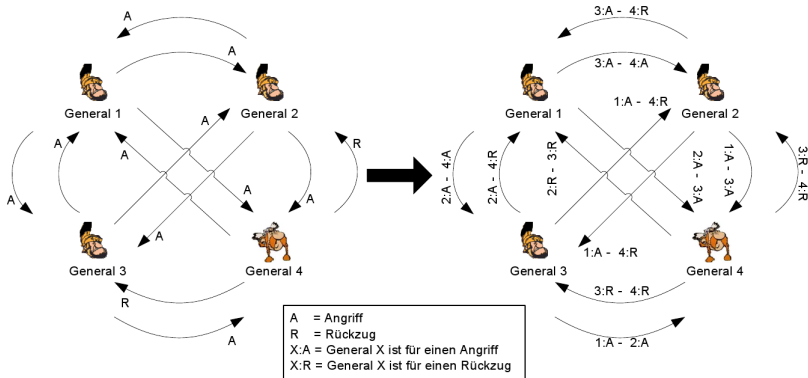
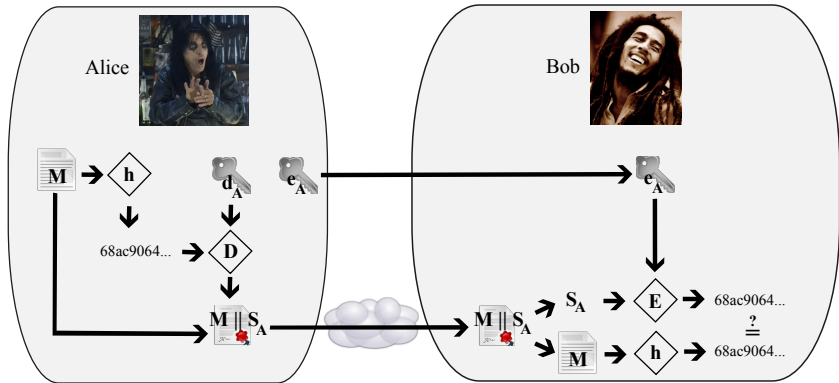
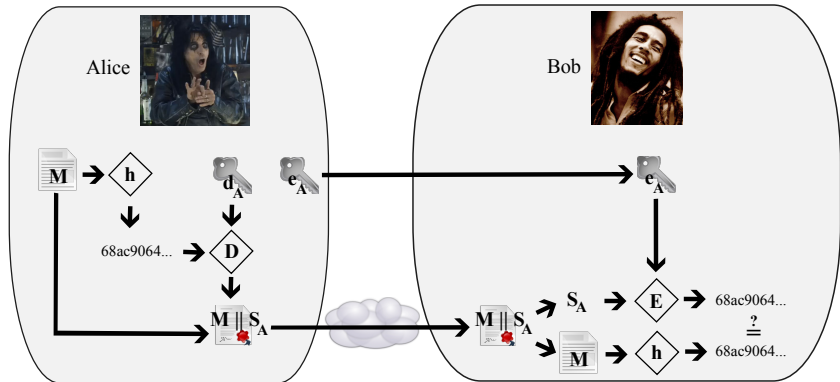


Abbildung 1: Byzantine Agreement Beispiel [2]

Public-Key-Signaturschema



Public-Key-Signaturschema



Wie kann man sicherstellen, dass der öffentliche Schlüssel e_A wirklich zu Alice gehört?

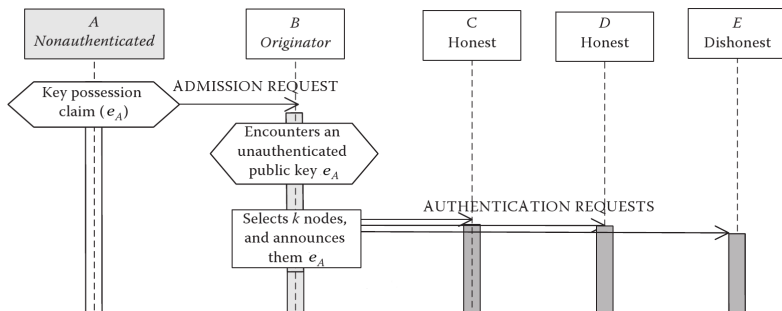
Problematik in dezentralen Netzen

- Knoten werden ständig dynamisch hinzugefügt und entfernt.
- Ohne eine Trusted Third Party (TTP) ist es nicht ohne weiteres möglich zu entscheiden ob ein neuer Knoten vertrauenswürdig ist und der angegebene öffentliche Schlüssel e_A wirklich zu diesem neuen Knoten A gehört.

Problematik in dezentralen Netzen

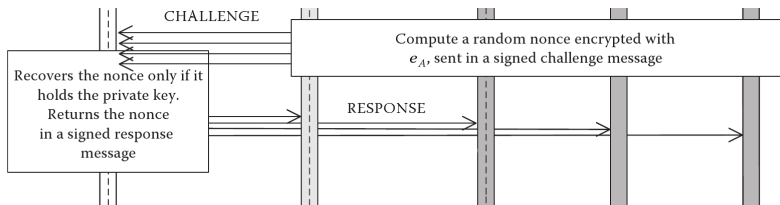
- Knoten werden ständig dynamisch hinzugefügt und entfernt.
- Ohne eine Trusted Third Party (TTP) ist es nicht ohne weiteres möglich zu entscheiden ob ein neuer Knoten vertrauenswürdig ist und der angegebene öffentliche Schlüssel e_A wirklich zu diesem neuen Knoten A gehört.
- *Lösungsansatz:* Authentisierungsprotokoll mit Byzantine Agreement („Byzantine fault tolerant public key authentication in peer-to-peer systems“ von Vivek Pathak und Liviu Iftode [3])

Authentisierungsprotokoll: 1. Verbindungsanfrage



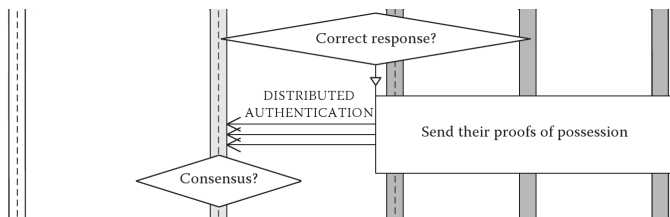
- Knoten B entdeckt den unauthorisierten Knoten A , der beitreten will.
- Knoten A stellt sich dem Netzwerk mit dem öffentlichen Schlüssel e_A vor.
- Knoten B initiiert eine gemeinsame Validierung von e_A durch die Gruppe der *vertrauenswürdigen Knoten* (C, D, E).
- Alle teilnehmenden Knoten bekommen e_A zugeschickt.

Authentisierungsprotokoll: 2. Challenge Response



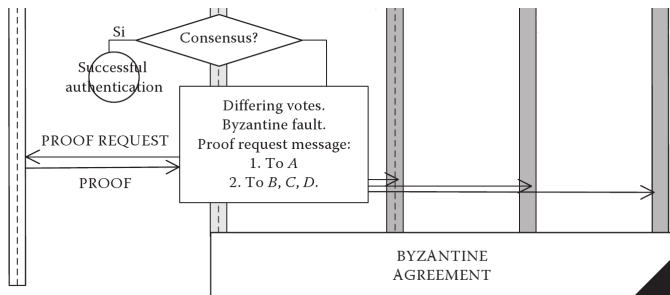
- Jeder teilnehmende Knoten generiert eine zufällige Nonce und chiffriert diese mit e_A .
- Die Knoten sendet jeweils ihre verschlüsselte Nonce an Knoten A.
- Knoten A muss nun jede empfangene Nonce mit dem privaten Schlüssel d_A entschlüsseln und zurücksenden.

Authentisierungsprotokoll: 3. Verteilte Authentisierung



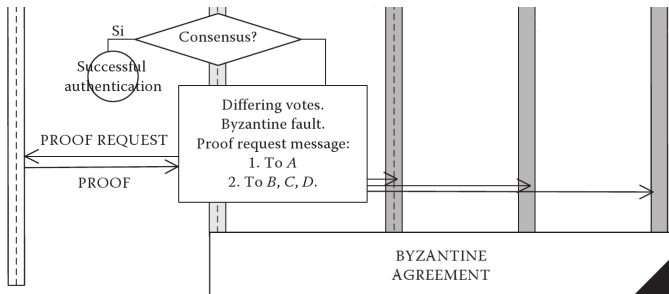
- Jeder Knoten vergleicht die empfangene Nonce mit der vorher verschlüsselt gesendeten Nonce.
 - Wenn sie übereinstimmen sendet jeder Knoten seinen Teil der Authentisierungszustimmung, die mit der Zustimmung von B die gesamte Authentisierung bilden.
 - Wenn keine Übereinstimmung vorliegt wird eine Authentisierungsablehnung an B geschickt.

Authentisierungsprotokoll: 4. Byzantine Agreement



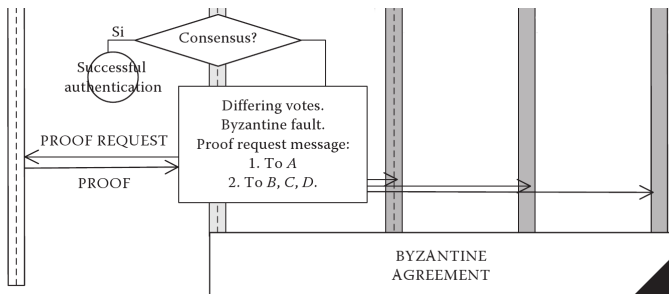
- Stimmen alle Knoten der Authentisierung zu wird Knoten *A* in die Gruppe der *vertrauenswürdigen Knoten* aufgenommen.
- Sobald ein Knoten der Authentisierung nicht zustimmt müssen weitere Schritte eingeleitet werden. → *Byzantine fault*

Authentisierungsprotokoll: 4. Byzantine Agreement



- Bei einem *Byzantine fault* schickt Knoten *B* nun einen *Proof Request* an *A*. *A* muss nun mit allen vorher empfangenen chiffrierten Noncen, und den dazugehörigen mit e_A dechiffrierten Noncen antworten.
- Ist Knoten *A* vertrauenswürdig, antwortet er mit den erwarteten Noncenpaaren. Somit ist aber mindestens ein anderer Knoten kompromittiert oder fehlerhaft.

Authentisierungsprotokoll: 4. Byzantine Agreement



- Nun wird von *B* ein *Byzantine fault* an alle anderen Knoten gesendet.
- Dadurch wird ein *Byzantine Agreement* Algorithmus eingeleitet, um eine gemeinsame Entscheidung bezüglich *A* zu treffen.

Fragen

Fragen?

Literatur



Leslie Lamport, Robert Shostak, and Marshall Pease.

The Byzantine Generals Problem.

ACM Transactions on Programming Languages and Systems, 4:382–401, 1982.



Timo Müller.

Das byzantinische Abkommen.

Seminararbeit, Fachbereich Medieninformatik, Hochschule der Medien Stuttgart, February 2008.



Vivek Pathak and Liviu Iftode.

Byzantine fault tolerant public key authentication in peer-to-peer systems.

Computer Networks, 50(4):579–596, 2006.



Yan Zhang, Hsiao-Hwa Chen, and Mohsen Guizani.

Cooperative Wireless Communications.

Auerbach Publications, 2009.

Authentisierungsprotokoll in dezentralen Netzen

- Authentisierungsalgorithmus nach „Byzantine fault tolerant public key authentication in peer-to-peer systems“ von Vivek Pathak und Liviu Iftode [3].

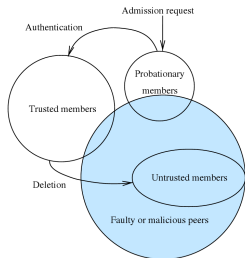


Abbildung 2: Gruppeneinteilung [3]

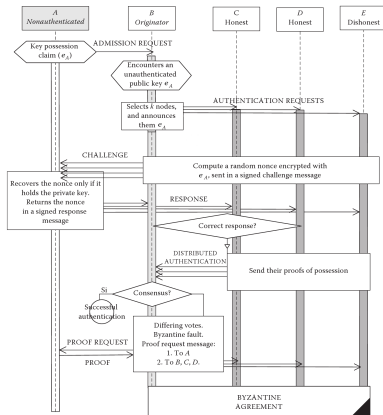


Abbildung 3: Sequenzdiagramm [4]

Byzantine Agreement

- Beispiel mit $n = 4$ Generälen und $f = 1$ Verrätern in einem vollständigem Graphen $G = (V, E)$: (Lösbar wenn: $n > 3f$ und $\forall i \in V : \delta(i) = 2f$)

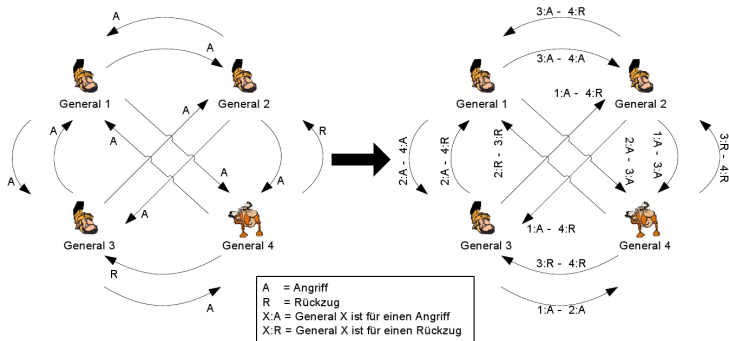


Abbildung 4: Byzantine Agreement Beispiel [2]