CoNe Freiburg

Peer-to-Peer Networks

Security
10th Week

Albert-Ludwigs-Universität Freiburg
Department of Computer Science
Computer Networks and Telematics
Christian Schindelhauer
Summer 2008

Attacks

- **Denial-of-Service Attacks (DoS)**
  - or distributed denial of service attacks (DDoS)
  - one or many peers ask for a document
  - peers are slowed down or blocked completely
- **Sybil Attacks**
  - one attacker produces many fake peers under new IP addresses
  - or the attacker controls a bot-net
- **Use of protocol weaknesses**
- **Infiltration by malign peers**
  - Byzantine Generals
- **Timing attacks**
  - messages are slowed down
  - communication line is slowed down
  - a connection between sender and receiver can be established
- **Poisoning Attacks**
  - provide false information
  - wrong routing tables, wrong index files etc.
- **Eclipse Attack**
  - attack the environment of a peer
  - disconnect the peer
  - build a fake environment

Solutions to the Sybil Attack

- Survey paper by Levine, Shields, Margonin, 2006
- **Trusted certification**
  - only approach to completely eleminate Sybil attacks
    - according to Douceur
  - relies on centralized authority
- **No solution**
  - know the problem and deal with the consequences
- **Resource testing**
  - real world friends
  - test for real hardware or addresses
    - e.g. heterogeneous IP addresses
  - check for storing ability
- **Recurring cost and fees**
  - give the peers a periodic task to find out whether there is real hardware behind each peer
    - wasteful use of resources
  - charge each peer a fee to join the network
- **Trusted devices**
  - use special hardware devices which allow to connect to the network

Solutions to the Sybil Attack

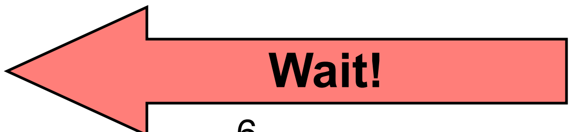
- Survey paper by Levine, Shields, Margonin, 2006
- **In Mobile Networks**
  - use observations of the mobile node
    - e.g. GPS location, neighbor nodes, etc.
- **Auditing**
  - perform tests on suspicious nodes
  - or reward a peer who proves that it is not a clone peer
- **Reputation Systems**
  - assign each peer a reputation which grows over the time with each positive fact
  - the reputation indicates that this peer might behave nice in the future
  - Disadvantage:
    - peers might pretend to behave honestly to increase their reputation and change their behavior in certain situations
    - problem of Byzantine behavior

The Problem of Byzantine Generals

- **3 armies prepare to attack a castle**
- **They are separated and communicate by messengers**
- **If one army attacks alone, it loses**
- **If two armies attack, they win**
- **If nobody attacks the castle is besieged and they win**
- **One general is a renegade**
  - nobody knows who

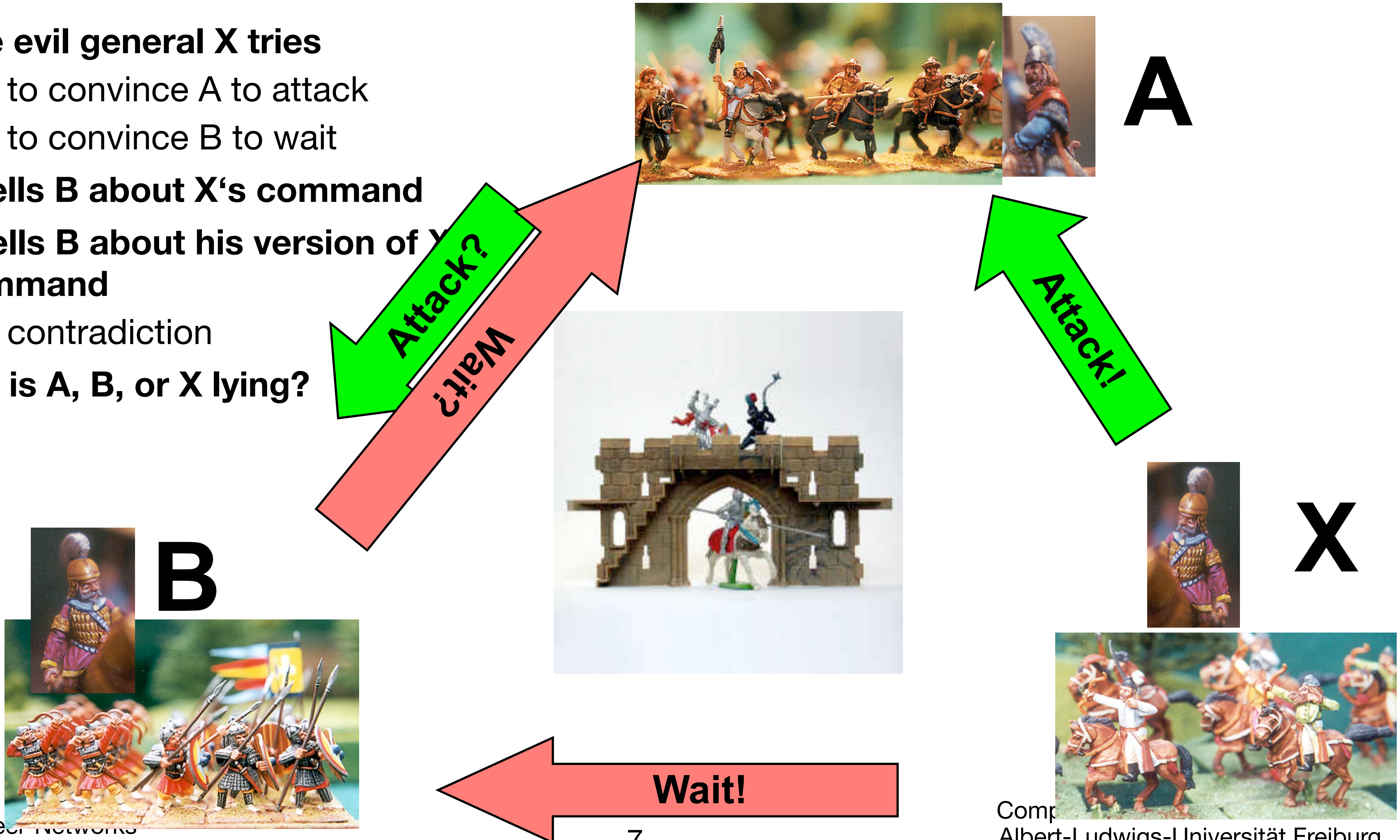
The Problem of Byzantine Generals

- **The evil general X tries**
  - to convince A to attack
  - to convince B to wait
- **A tells B about X‘s command**
- **B tells B about his version of X‘s command**
  - contradiction
- **But is A, B, or X lying?**



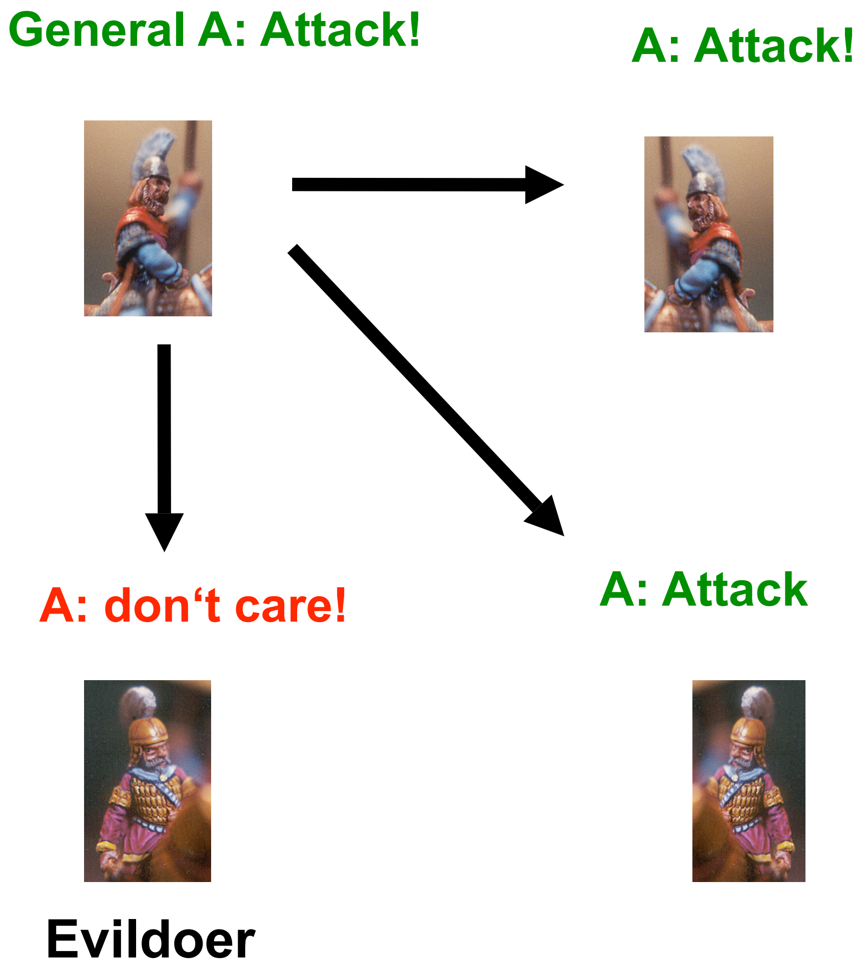
The Problem of Byzantine Generals

- **The evil general X tries**
  - to convince A to attack
  - to convince B to wait
- **A tells B about X‘s command**
- **B tells B about his version of X command**
  - contradiction
- **But is A, B, or X lying?**
  -



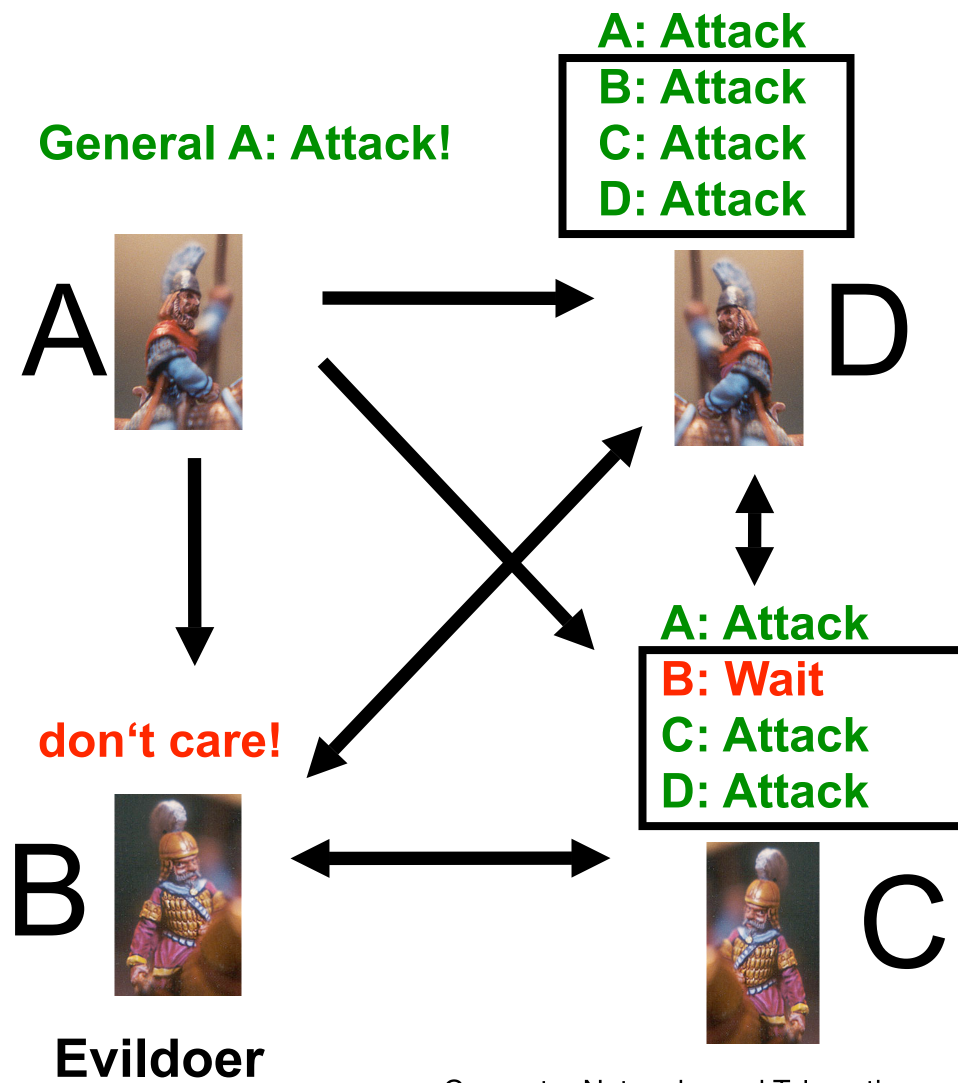
Byzantine Agreement

- **Theorem**
  - The problem of three byzantine generals cannot be solved (without cryptography)
  - It can be solved for 4 generals
- **Consider: 1 general, 3 officers problem**
  - If the general is loyal then all loyal officers will obey the command
  - In any case distribute the received commans to all fellow officers
  - What if the general is the renegade?



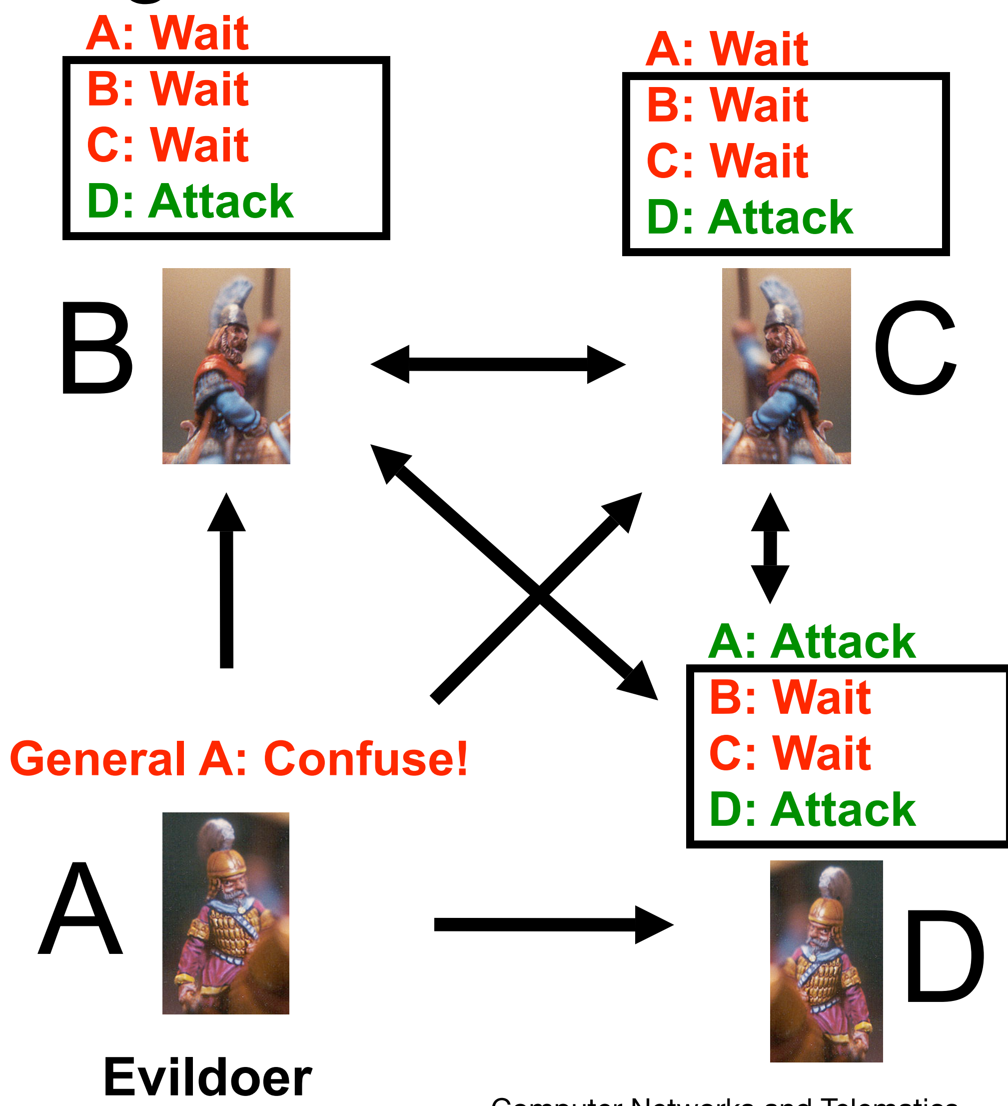
Byzantine Agreement

- **Theorem**
  - The problem of four byzantine generals can be solved (without cryptography)
- **Algorithm**
  - General A sends his command to all other generals
    - A sticks to his command if he is honest
  - All other generals forward the received command to all other generals
  - Every generals computes the majority decision of the received commands and follows this command



Byzantine Agreement

- **Theorem**
  - The problem of four byzantine generals can be solved (without cryptography)
- **Algorithm**
  - General A sends his command to all other generals
    - A sticks to his command if he is honest
  - All other generals forward the received command to all other generals
  - Every generals computes the majority decision of the received commands and follows this command



General Solution of Byzantine Agreement

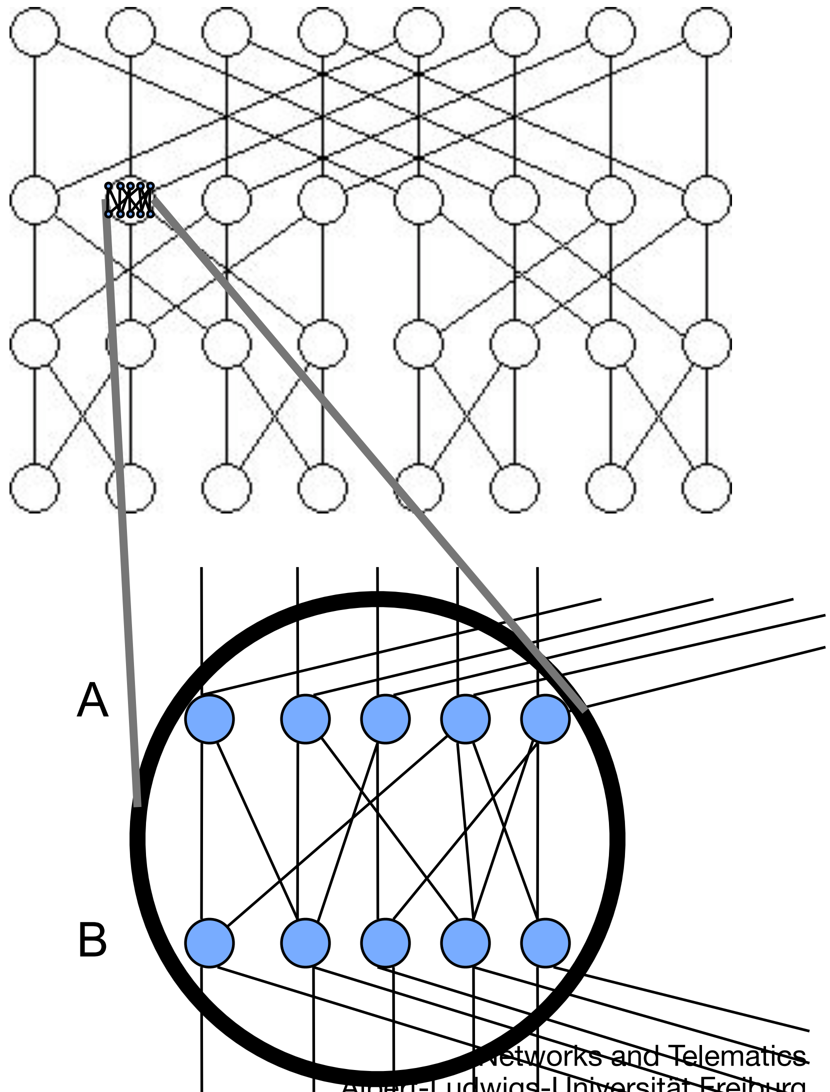
- **Theorem**
  - If m generals are traitors then 2m+1 generals must be honest to get a Byzantine Agreement
- **This bound is sharp if one does not rely on cryptography**
- **Theorem**
  - If a digital signature scheme is working, then an arbitrarily large number of betraying generals can be dealt with
- **Solution**
  - Every general signs his command
  - All commands are shared together with the signature
  - Inconsistent commands can be detected
  - The evildoer can be exposed

P2P and Byzantine Agreement

- **Digital signature can solve the problem of malign peers**
- **Problem: Number of messages**
  - $O(n^2)$ messages in the whole network (for n peers)
- **In „Scalable Byzantine Agreement“ von Clifford Scott Lewis und Jared Saia, 2003**
  - a scalable algorithm was presented
  - can deal with n/6 evil peers
    - if they do not influence the network structure
  - use only O(log n) messages per node in the expectation
  - find agreement with high probability

Network of Lewis and Saia

- **Butterfly network with clusters of size c log n**
  - clusters are bipartite expander graphs
  - Bipartite graph
    - is a graph with disjoint node sets A and B where no edges connect the nodes within A or within B
  - Expander graph
    - A bipartite graph is an expander graph if for each subset X of A the number of neighbors in B is at least c|X| for a fixed constant c>0
    - and vice versa for the subsets in B



Discussion

- **Advantage**
  - Very efficient, robust and simple method
- **Disadvantage**
  - Strong assumptions
    - The attacker does not know the internal network structure
- **If the attacker knows the structure**
  - Eclipse attack!

Cuckoo Hashing for Security

- **Awerbuch, Scheideler, Towards Scalable and Robust Overlay Networks**
- **Problem:**
  - Rejoin attacks
- **Solution:**
  - Chord network combined with
  - Cuckoo Hashing
  - Majority condition:
    - honest peers in the neighborhood are in the majority
  - Data is stored with O(log n) copies

Cuckoo Hashing

- **Collision strategy for (classical) hashing**
  - uses two hash functions h_1, h_2
  - an item with key x is either stored at $h_1(x)$ or $h_2(x)$
    - easy lookup
- **Insert x**
  - try inserting at $h_1(x)$ or $h_2(x)$
  - if both positions are occupied then
    - kick out one element
    - and insert it at its other place
    - continue this with the next element if the position is occupied

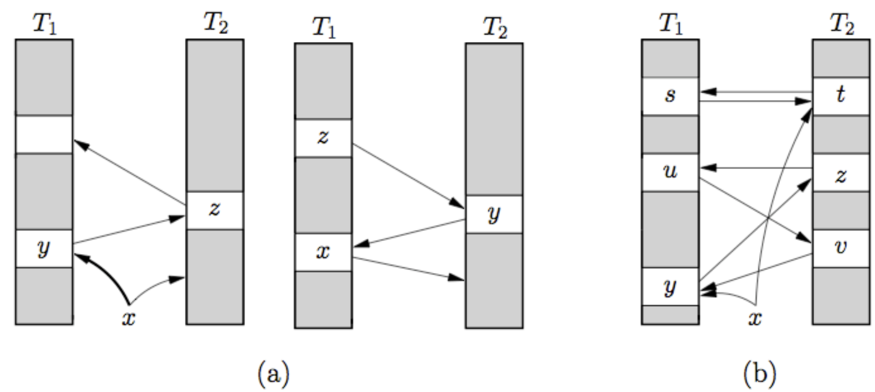


Fig. 1. Examples of CUCKOO HASHING insertion. Arrows show possibilities for moving keys. (a) Key x is successfully inserted by moving keys y and z from one table to the other. (b) Key x cannot be accommodated and a rehash is necessary.

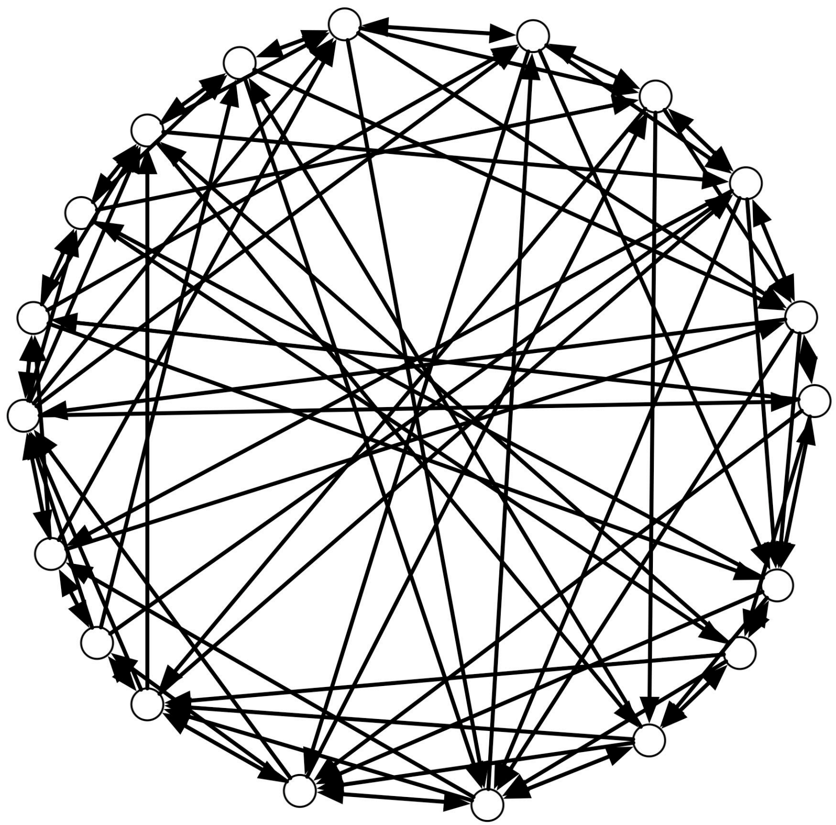
From Cuckoo Hashing
Rasmus Pagh, Flemming Friche Rodler
2004

Efficiency of Cuckoo Hashing

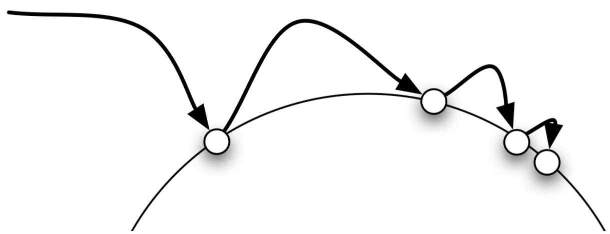
- **Theorem**
  - Let $\epsilon>0$ then if at most n elements are stored, then Cuckoo Hashing needs a hash space of $2n+\epsilon$.
- **Three hash functions increase the load factor from 1/2 to 91%**
- **Insert**
  - needs O(1) steps in the expectation
  - O(log n) with high probability
- **Lookup**
  - needs two steps

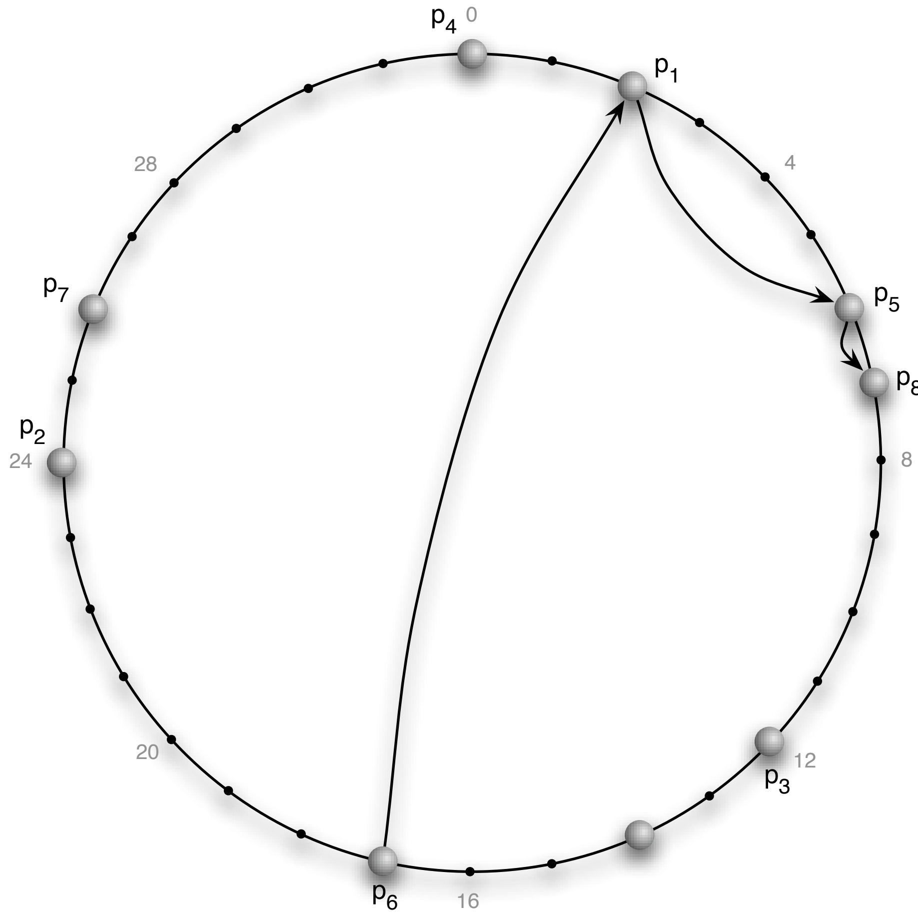
Chord

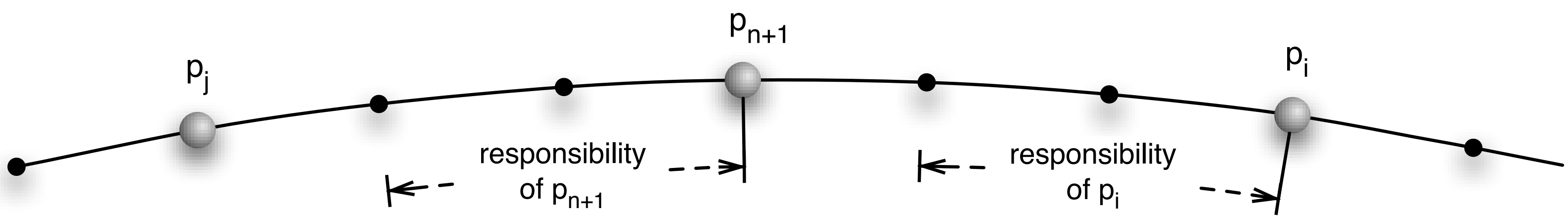
- **Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek and Hari Balakrishnan (2001)**
- **Distributed Hash Table**
  - range $\{0,..,2^m-1\}$
  - for sufficient large m
- **for this work the range is seen as [0,1)**
- **Network**
  - ring-wise connections
  - shortcuts with exponential increasing distance



Lookup in Chord

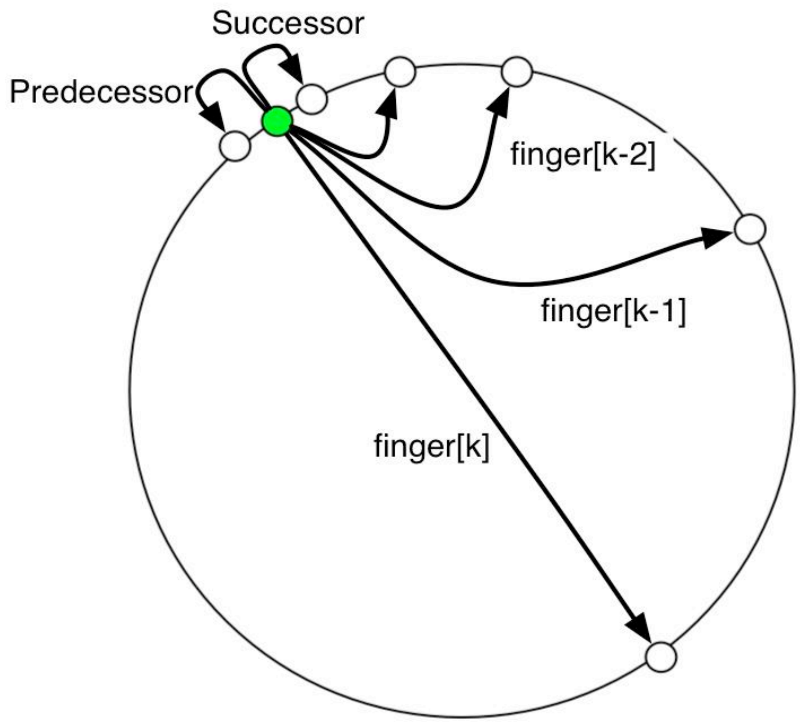






Data Structure of Chord

- **For each peer**
  - successor link on the ring
  - predecessor link on the ring
  - for all $i \in \{0,..,m-1\}$
    - Finger[i] := the peer following the value $r_V(b+2^i)$s
- **For small i the finger entries are the same**
  - store only different entries
- **Chord**
  - needs O(log n) hops for lookup
  - needs $O(\log^2 n)$ messages for inserting and erasing of peers



Cuckoo Hashing for Security

- **Given n honest peers and ϵ n dishonest peers**
- **Goal**
  - For any adversarial attack the following properties for every interval $I \subseteq [0, 1)$ of size at least (c log n) we have
  - Balancing condition
    - I contains $\Theta(|I| \cdot n)$ nodes
  - Majority condition
    - the honest nodes in I are in the majority
- **Then all majority decisions of O(log n) nodes give a correct result**

Rejoin Attacks

- **Secure hash functions for positions in the Chord**
  - if one position is used
  - then in an O(log n) neighborhood more than half is honest
  - if more than half of al peers are honest
- **Rejoin attacks**
  - use a small number of attackers
  - check out new addresses until attackers fall in one interval
  - then this neighborhood can be ruled by the attackers

The Cuckoo Rule for Chord

- **Notation**
  - a region is an interval of size $1/2^r$ in [0, 1) for some integer r that starts at an integer multiple of $1/2^r$
  - There are exactly 2^r regions
  - A k-region is a region of size (closest from above to) k/n, and for any point $x \in [0, 1)$
  - the k-region $R_k(x)$ is the unique k-region containing x.
- **Cuckoo rule**
  - If a new node v wants to join the system, pick a random $x \in [0, 1)$.
  - Place v into x and move all nodes in $R_k(x)$ to points in [0, 1) chosen uniformly at random
    - (without replacing any further nodes).
- **Theorem**
  - For any constants ϵ and k with $\epsilon < 1-1/k$, the cuckoo rule with parameter k satisfies the balancing and majority conditions for a polynomial number of rounds, with high probability, for any adversarial strategy within our model.
  - The inequality $\epsilon < 1 - 1/k$ is sharp

Operations

- **Data storage**
  - each data item is stored in the $O(\log^3 n)$ neighborhood as copies
- **Primitives**
  - robust hash functions
    - safe against attacks
  - majority decisions of each operation
  - use multiple routes for targeting location

Efficiency

- **Lookup**
  - works correctly with high probability
  - can be performed with $O(\log^5 n)$ messages
- **Inserting of data**
  - works in polylogarithmic time
  - needs $O(\log^5 n)$ messages
- **Copies stored of each data: $O(\log^3 n)$**

Discussion

- **Advantage**
  - Cuckoo Chord is safe against adversarial attacks
  - Cuckoo rule is simple and effective
- **Disadvantage**
  - Computation of secure hash function is complex
  - Considerate overhead for communication
- **Theoretical breakthrough**
- **Little impact to the practical world**

CoNe
Freiburg

Peer-to-Peer Networks

End of 10th Week

Albert-Ludwigs-Universität Freiburg
Department of Computer Science
Computer Networks and Telematics
Christian Schindelhauer
Summer 2008