Peer-to-Peer Networks

14 Security

Christian Ortolf
Technical Faculty
Computer-Networks and Telematics
University of Freiburg

Motivation for Anonymity

- Society
  - Free speech is only possible if the speaker does not suffer negative consequences
  - Thus, only an anonymous speaker has truly free speech
- Copyright infringement
  - Copying items is the best (and most) a computer can do
  - Copyright laws restrict copying
  - Users of file sharing systems do not want to be penalized for their participation or behavior
- Dictatorships
  - A prerequisite for any oppressing system is the control of information and opinions
  - Authors, journalists, civil rights activists like all citizens should be able to openly publish documents without the fear of penalty
- Democracies
  - Even in many democratic states certain statements or documents are illegitimate, e.g.
    - (anti-) religious statements
    - insults (against the royalty)
    - certain types of sexual contents
    - political statements (e.g. for fascism, communism, separation, revolution)
- A anonymizing P2P network should secure the privacy and anonymity of each user without endangering other users

Terms

- From
  - Danezis, Diaz, A Survey of Anonymous Communication Channels
  - Pfitzmann, Hansen, Anonymity, Unobservability and Pseudonymity – A Proposal for Terminology
- Anonymity (Pfitzmann-Hansen 2001)
  - describes the state of being not identifiable within a larger set of subjects (peers), i.e.
    - the anonymity set
  - The anonymity set can be all peers of a peer-to-peer network
    - yet can be another (smaller or larger) set

Terms

- Unlinkability
  - Absolute (ISO15408)
    - „ensures that a user may make multiple uses of resources or services without other being able to link these uses together.“
  - Relative
    - Any attacker cannot find out more about the connections of the uses by observing the system
      - a-priori knowledge = a-posteriori knowledge

Terms

- Unobservability
  - The items of interests are protected
  - The use or non-use of any service cannot be detected by an observer (attacker)
- Pseudonymity
  - is the use of pseudonyms as IDs
  - preserves accountability and trustability while preserving anonymity

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

Attacks

- 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
- Surveillance
  - full or partial

Cryptography in a Nutshelf

- Symmetric Cryptography
  - AES
  - Affine Cryptosystems
- Public-Key Cryptography
  - RSA
  - ElGamal
- Digital Signatures
- Public-Key-Exchange
  - Diffie-Hellman
- Interactive Proof Systems
  - Zero-Knowledge-Proofs
  - Secret Sharing
  - Secure Multi-Party Computation

Blakley's Secret Sharing

- George Blakley, 1979
- Task
  - n persons have to share a secret
  - only when k of n persons are present the secret is allowed to be revealed
- Blakley's scheme
  - in a k-dimensional space the intersection of k non-parallel k-1-dimensional spaces define a point
  - this point is the information
  - with k-1 sub-spaces one gets only a line
- Construction
  - A third (trusted) instance generate for a point n in R^k k non-parallel k-1-dimensional hyper-spaces

Shamir's Secret Sharing Systems

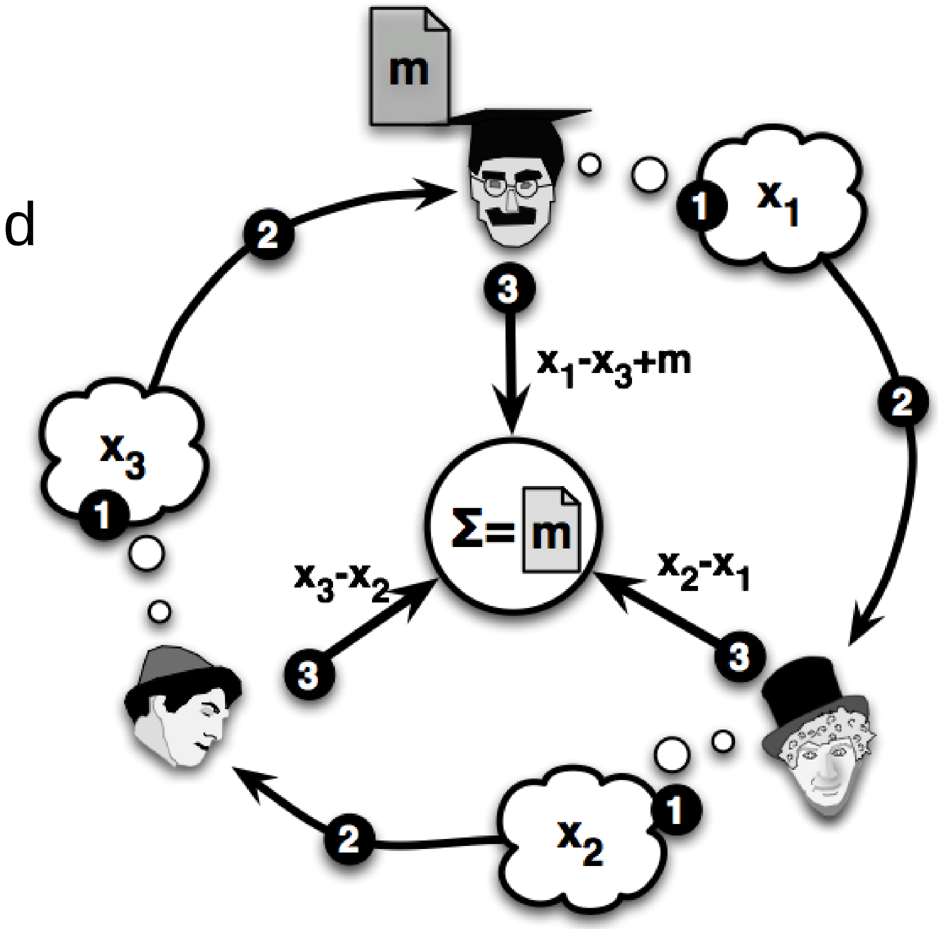
- Adi Shamir, 1979
- Task
  - n persons have to share a secret s
  - only k out of n persons should be able to reveal this secret
- Construction of a trusted third party
  - chooses random numbers $a_1,...,a_{k-1}$
  - defines
    $$f(x) = s + a_1 x + a_2 x^2 + \ldots + a_{k-1} x^{k-1}$$
  - chooses random $x_1, x_2, ..., x_n$
  - sends $(x_i, f(x_i))$ to player i

Shamir‘s Secret Sharing Systems

- If k persons meet
  - then they can compute the function f by the fundamental theorem of algebra
    - a polynomial of degree d is determined by d+1 values
  - for this they exchange their values and compute by interpolation
    - (e.g. using Lagrange polynoms)
- If k-1 persons meet
  - they cannot compute the secret at all
  - every value of s remains possible
- Usually, Shamir‘s and Blakley‘s scheme are used in finite fields
  - i.e. Galois fields (known from CRC)
  - this simplifies the computation and avoids rounding errors in the context of floating numbers

Dining Cryptographers

- Anonymous publications without any tracing possibility
- $n \geq 3$ cryptographers sit at a round table
- neighbored cryptographers can communicate secretly
- Each peer chooses secret number x_i and communicates it to the right neighbor
- If i wants to send a message m
- he publishes $s_i = x_i - x_{i-1} + m$
- else
- he publishes $s_i = x_i - x_{i-1}$
- Now they compute the sum $s=s_1+...+s_n$
- if s=0 then there is no message
- else the sum of all messages



Encryption Methods

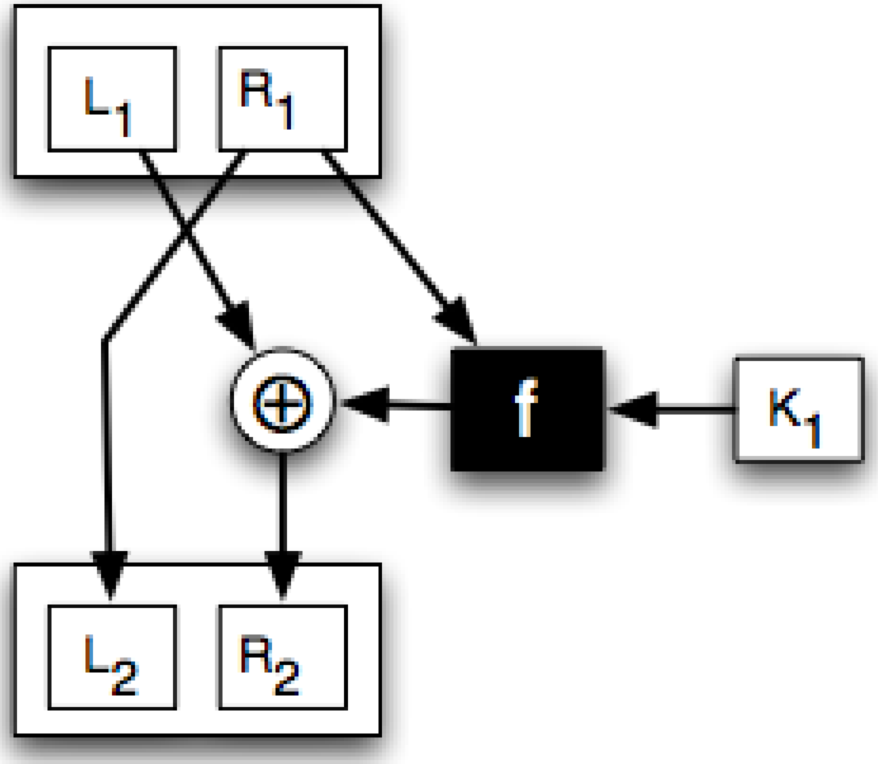
- Symmetric encryption algorithms, e.g.
  - Feistel cipher
  - DES (Digital Encryption Standard)
  - AES (Advanced Encryption Standard)
- Cryptographic hash function
  - SHA-1, SHA-2
  - MD5
- Asymmetric encryption
  - RSA (Rivest, Shamir, Adleman)
  - El-Gamal
- Digital signatures (electronic signatures)
  - PGP (Phil Zimmermann), RSA

Symmetric Encryption

- E.g. Caesar's code, DES, AES
- Functions f and g, where
  - Encryption f
    - f (key, text) = code
  - Decoding g:
    - g (key, code) = text
- The key
  - must remain secret
  - must be available to the sender and receiver

Feistel Chiffre

- Splitting the message into two halves L_1, R_1
  - Keys K_1, K_2, ...
  - Several rounds: Resulting code: L_n, R_n
- encoding
  - $L_i = R_{i-1}$
  - $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$
- Decryption
  - $R_{i-1} = L_i$
  - $L_{i-1} = R_i \oplus f(L_i, K_i)$
- f may be any complex function



Other Symmetric Codes

- Skipjack
  - 80-bit symmetric code
  - is based on Feistel Cipher
  - low security
- RC5
  - 1-2048 bits key length
  - Rivest code 5 (1994)
  - Several rounds of the Feistel cipher

Digital Encryption Standard

- Carefully selected combination of
  - Xor operations
  - Feistel cipher
  - permutations
  - table lookups
  - used 56-bit key
- 1975 developed at IBM
  - Now no longer secure
  - more powerful computers
  - New knowledge in cryptology
- Succeeded by: AES (2001)

Advanced Encryption Standard

- Carefully selected combination of
  - Xor operations
  - Feistel cipher
  - permutations
  - table lookups
  - multiplication in GF $[2^8]$
  - 128, 192 or 256-bit symmetric key
- Joan Daemen and Vincent Rijmen
  - 2001 were selected as AES, among many
  - still considered secure

Cryptographic Hash Function

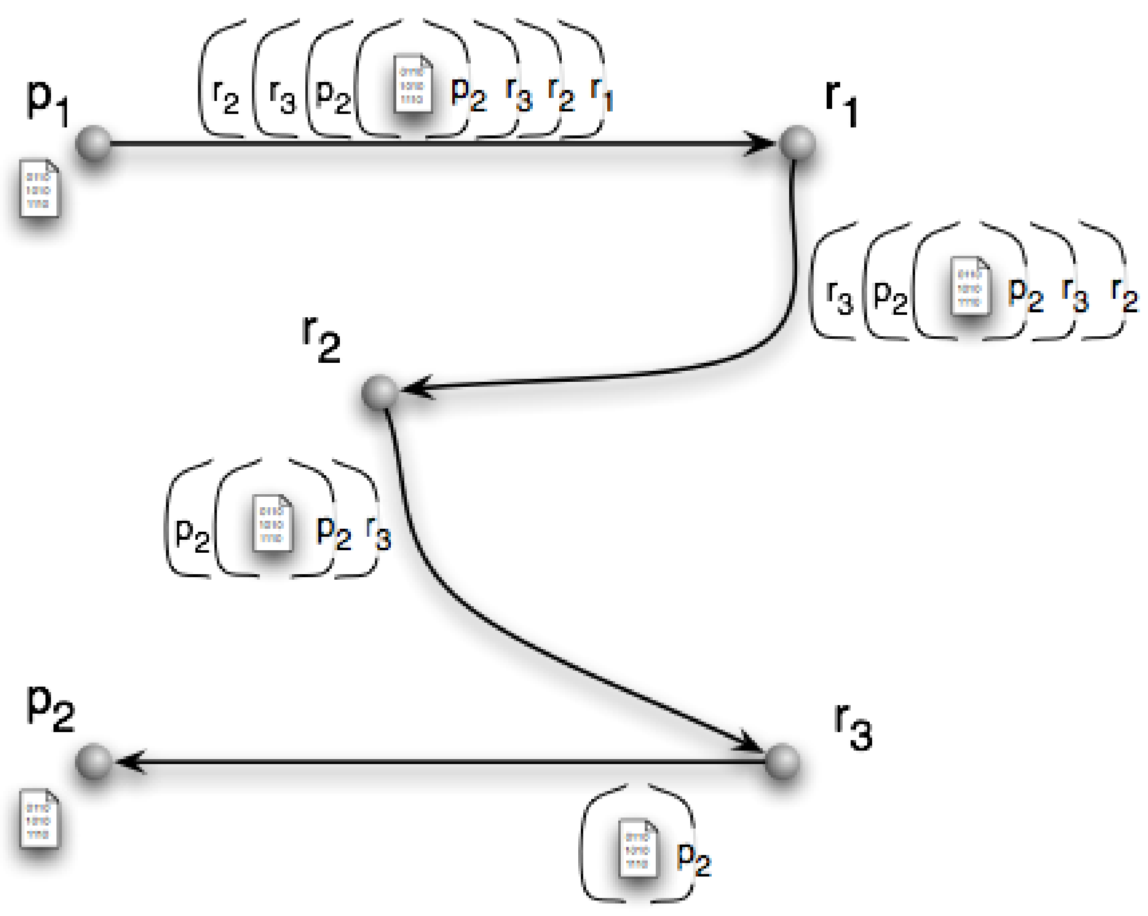
- E.g. SHA-1, SHA-2, MD5
- A cryptographic hash function h maps a text to a fixed-length code, so that
  - h(text) = code
  - it is impossible to find another text:
    - h(text‘) = h(text) and text ≠ text'
- Possible solution:
  - Using a symmetric cipher

Asymmetric Encryption

- E.g. RSA, Ronald Rivest, Adi Shamir, Lenard Adleman, 1977
  - Diffie-Hellman, PGP
- Secret key: sk
  - Only the receivers of the message know the secret key
- Public key: pk
  - All participants know this key
- Generated by
  - keygen(sk) = pk
- Encryption function f and decryption function g
  - Known to everybody
- Encryption
  - f(pk,text) = code
  - everybody can generate code
- Decryption
  - g(sk,code) = code
  - only possibly by receiver

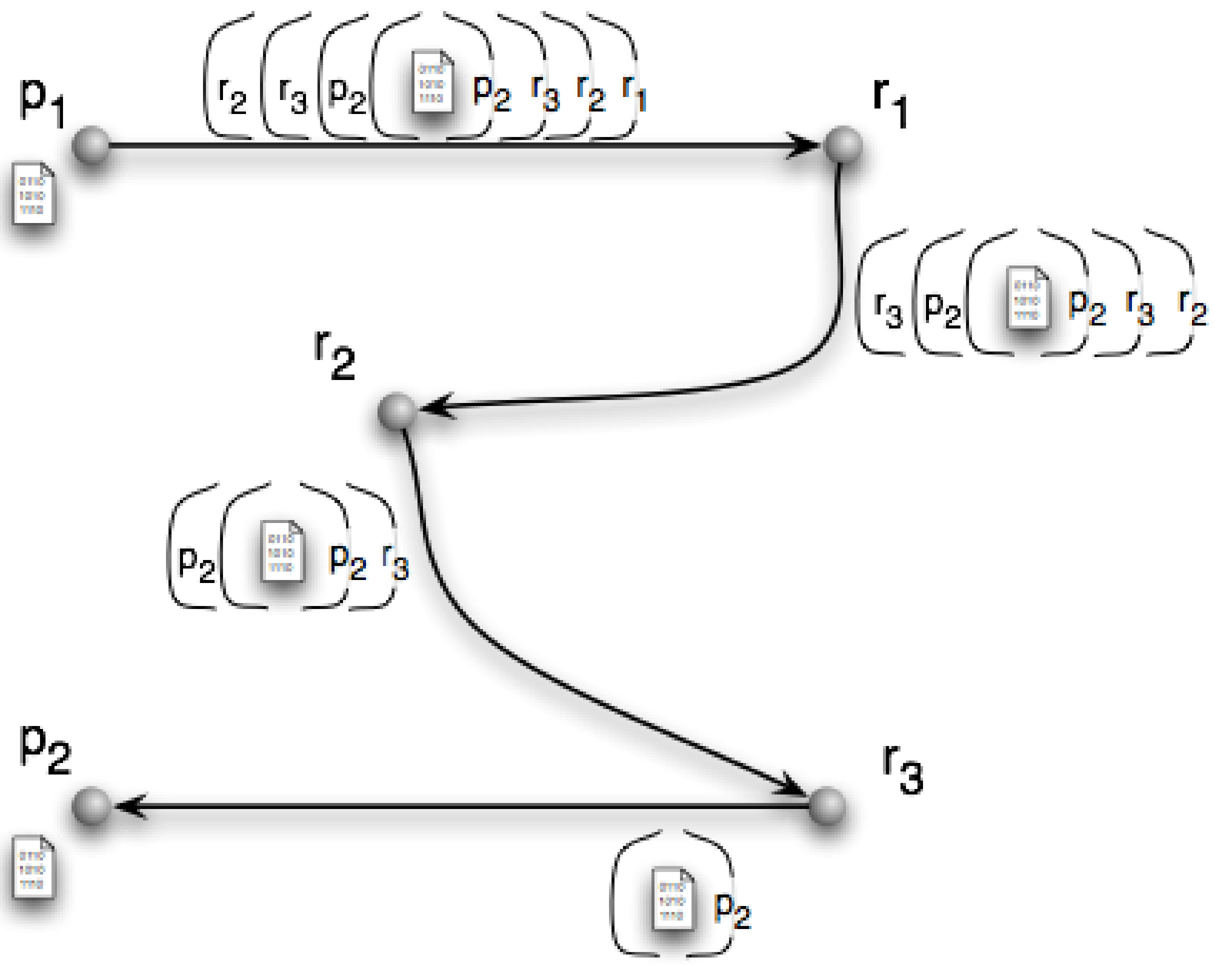
Chaum's Mix-Cascades

- All peers
  - publish the public keys
  - are known in the network
- The sender p_1 now chooses a route
  - p_1, r_1, r_2, r_3, ..., p_2
- The sender encrypts m according to the public keys from
  - p_2, ... r_3, r_2, r_1
  - and sends the message
  - $f(pk_{k1},(r_2,f(pk_{r2}...f(pk_{rk},(p_2,f(pk_{p2},m)))...)))))$
  - to r_1
- r_1 encrypts the code, deciphers the next hop r_2 and sends it to him
- ...
- until p_2 receives the message and deciphers it



Chaum's Mix Cascades

- No peer on the route
  - knows its position on the route
  - can decrypt the message
  - knows the final destination
- The receiver does not know the sender
- In addition peers may voluntarily add detour routes to the message
- Chaum's Mix Cascades
  - aka. Mix Networks or Mixes
  - is safe against all sort of attacks,
  - but not against traffic analysis



TOR - Onion Routers

- David Goldschlag, Michael Reed, and Paul Syverson, 1998
- Goal
  - Preserve private sphere of sender and receiver of a message
  - Safety of the transmitted message
- Prerequisite
  - special infrastructure (Onion Routers)
    - all except some smaller number of exceptions cooperate

TOR - Onion Routers

- Method
  - Mix Cascades (Chaum)
  - Message is sent from source to the target using proxies (Onion Routers)
  - Onion Routers unpredictably choose other routers as intermediate routers
  - Between sender, Onion Routers, and receiver the message is encrypted using symmetric cryptography
  - Every Onion Router only knows the next station
  - The message is encoded like an onion
- TOR is meant as an infrastructure improvement of the Internet
  - not meant as a peer-to-peer network
  - yet, often used from peer-to-peer networks

Other Work based on Onion Routing

- Crowds
  - Reiter & Rubin 1997
  - anonymous web-surfing based on Onion Routers
- Hordes
  - Shields, Levine 2000
  - uses sub-groups to improve Onion Routing
- Tarzan
  - Freedman, 2002
  - A Peer-to-Peer Anonymizing Network Layer
  - uses UDP messages and Chaum Mixes in group to anonymize Internet traffic
  - adds fake traffic against timing attacks

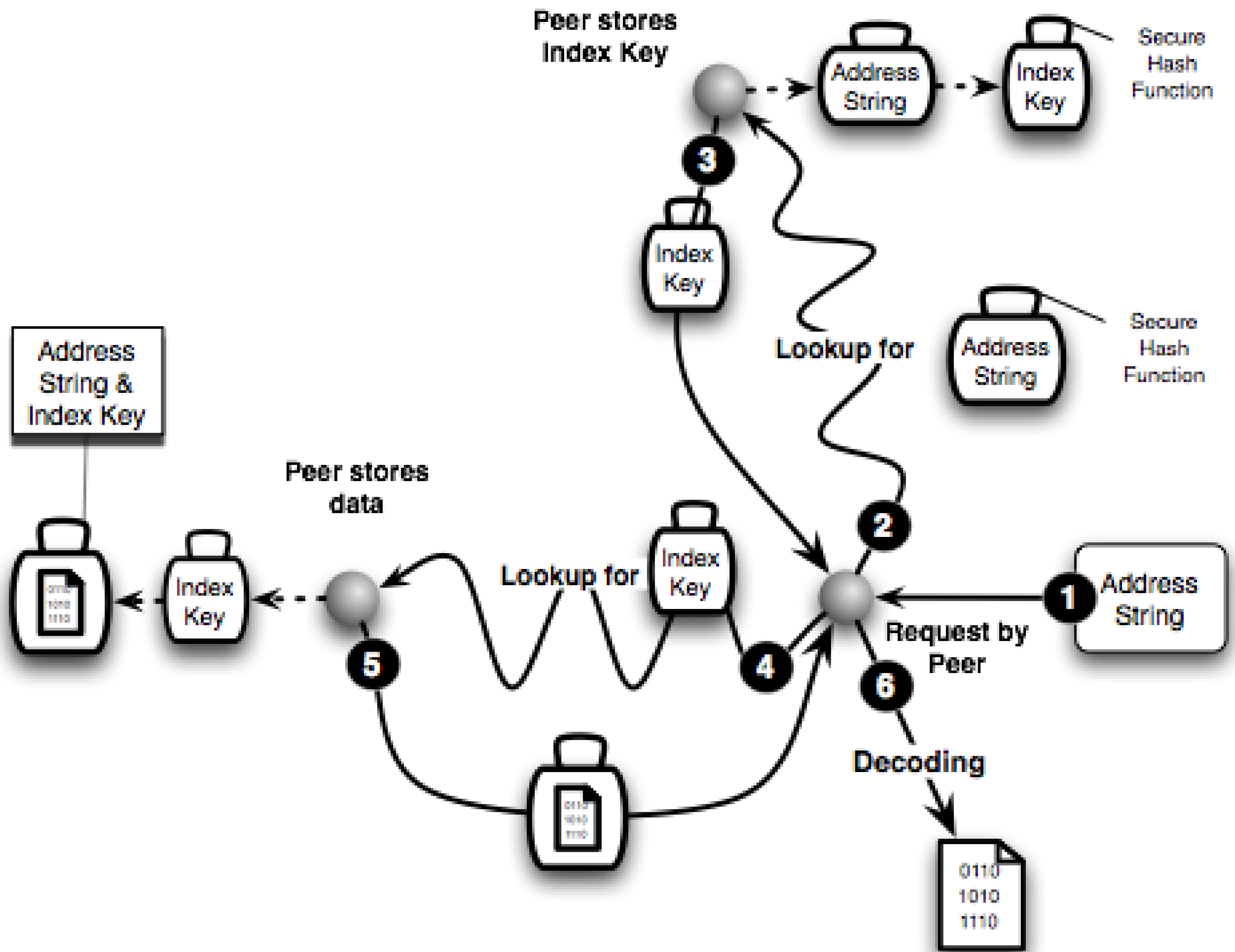
Free-Net

- Ian Clarke, Oskar Sandberg, Brandon Wiley, Theodore Hong, 2000
- Goal
  - peer-to-peer network
  - allows publication, replication, data lookup
  - anonymity of authors and readers
- Files
  - are encoding location independent
    - by encrypted and pseudonymously signed index files
    - author cannot be identified
  - are secured against unauthorized change or deletion
  - are encoded by keys unknown by the storage peer
    - secret keys are stored elsewhere
  - are replicated
    - on the look up path
  - and erased using “Least Recently Used” (LRU) principle

Free-Net

- Network Structure
  - is similar to Gnutella
  - Free-Net is like Gnutella Pareto distributed
- Storing Files
  - Each file can be found, decoded and read using the encoded address string and the signed subspace key
  - Each file is stored together with the information of the index key but without the encoded address string
  - The storage peer cannot read his files
    - unless he tries out all possible keywords (dictionary attack)
- Storing of index files
  - The address string coded by a cryptographic secure hash function leads to the corresponding peer
    - who stores the index data
      - address string
      - and signed subspace key
  - Using this index file the original file can be found

Free-Net



Peer stores
Index Key
Address
String
Index
Key
Secure
Hash
Function
3
Index
Key
Lookup for
Address
String
Secure
Hash
Function
Address
String &
Index Key
Peer stores
data
2
Index
Key
Lookup for
Index
Key
1
Address
String
Request by
Peer
5
4
6
Decoding
0110
1010
1110

Free-Net

- Lookup
  - steepest-ascent hill-climbing
    - lookup is forwarded to the peer whose ID is closest to the search index
  - with TTL field
    - i.e. hop limit
- Files are moved to new peers
  - when the keyword of the file is similar to the neighbor‘s ID
- New links
  - are created if during a lookup close similarities between peer IDs are discovered

Efficiency of Free-Net

- Network structure of Free-Net is similar to Gnutella
- The lookup time is polynomial on the average

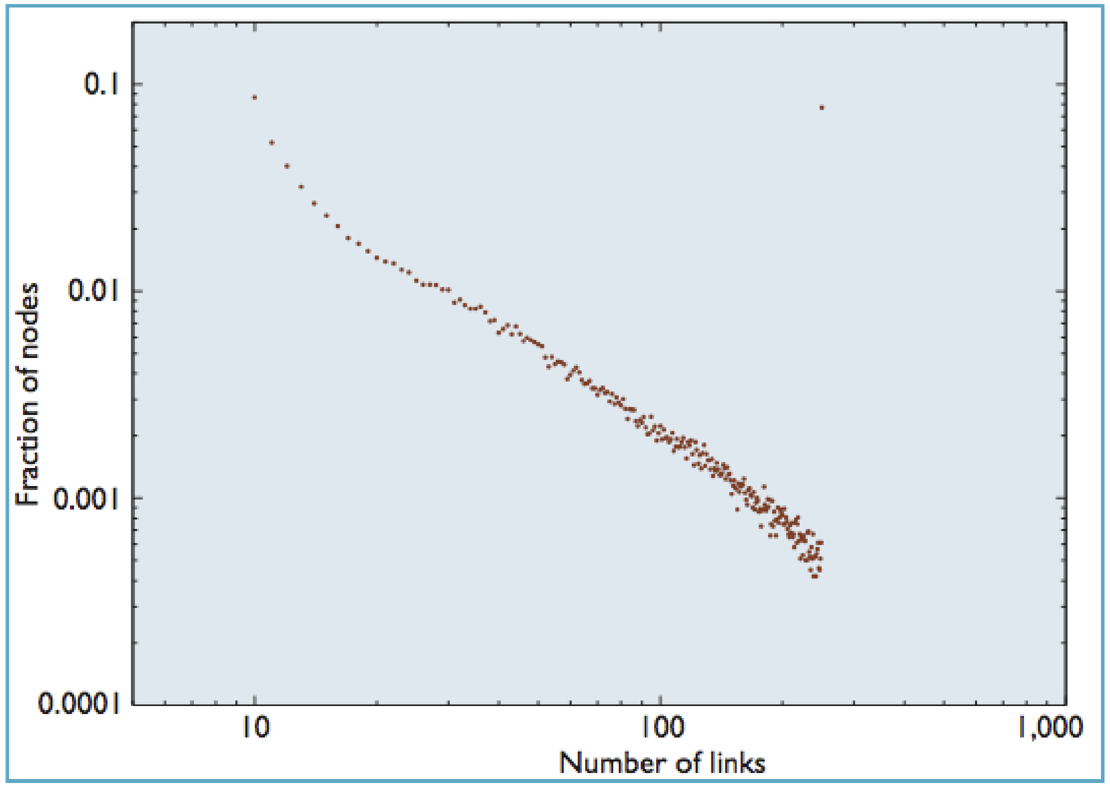


Figure 2. Degree distribution among Freenet nodes. The network shows a close fit to a power-law distribution.

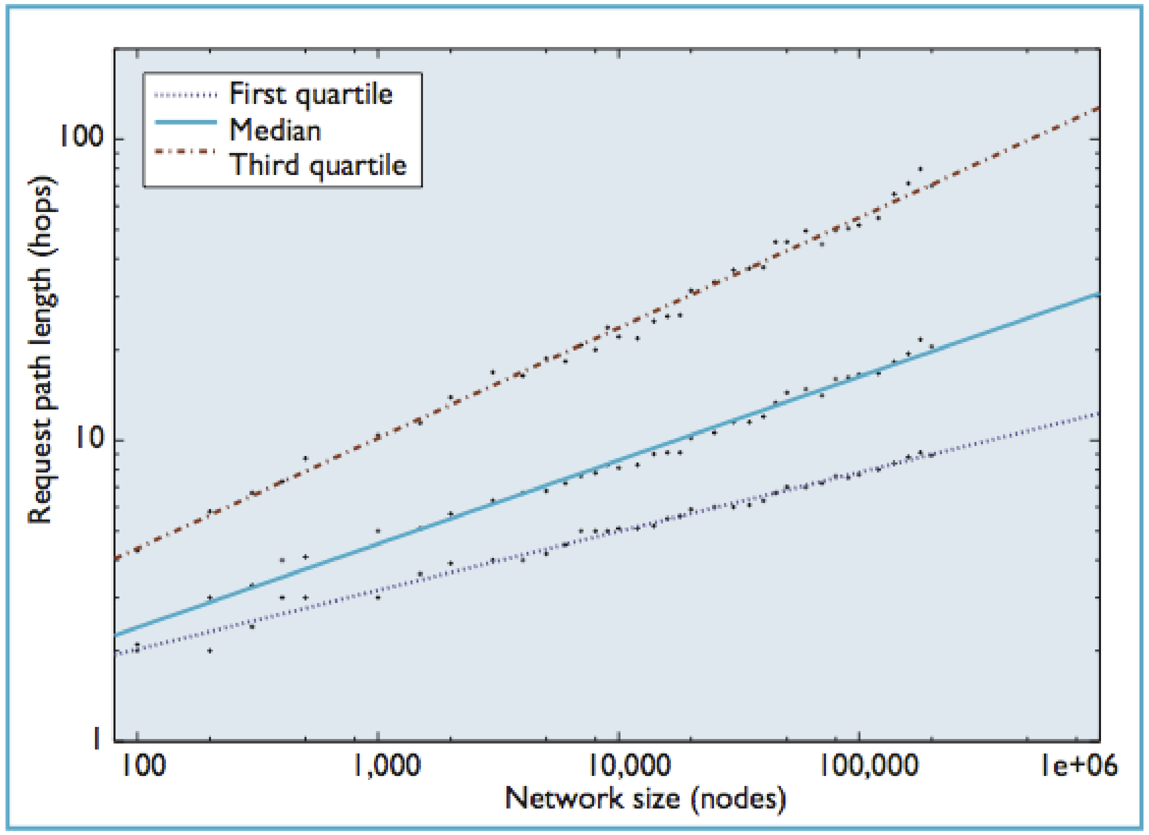


Figure 3. Request path length versus network size. The median path length in the network scales as $N^{0.28}$.

Dark-Net & Friend-to-Friend

- Dark-Net is a private Peer-to-Peer Network
  - Members can trust all other members
  - E.g.
    - friends (in real life)
    - sports club
- Dark-Net control access by
  - secret addresses,
  - secret software,
  - authentication using password, or
  - central authentication
- Example:
  - WASTE
    - P2P-Filesharing up to 50 members
    - by Nullsoft (Gnutella)
  - CSpace
    - using Kademlia

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

CoNe Freiburg

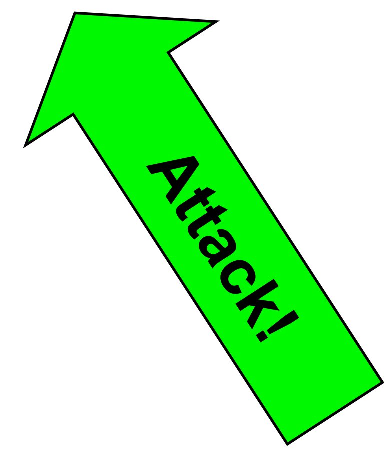
The Problem of Byzantine Generals

- 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

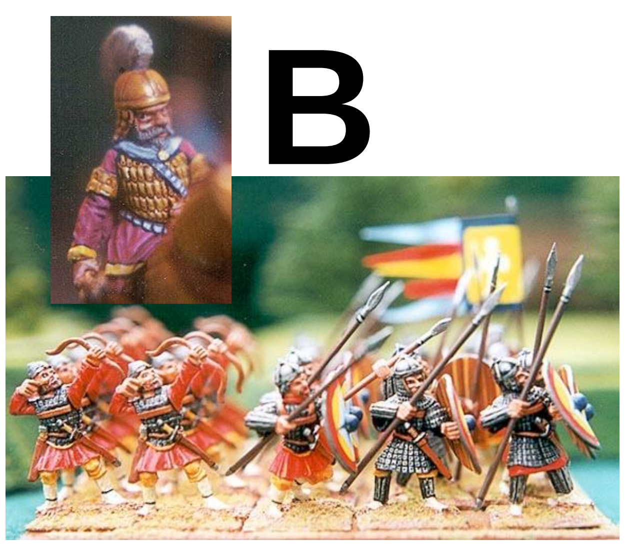
The Problem of Byzantine Generals

- 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?

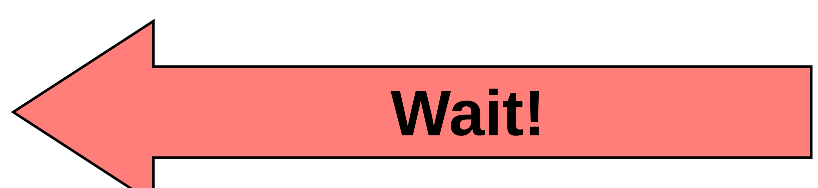
A



X

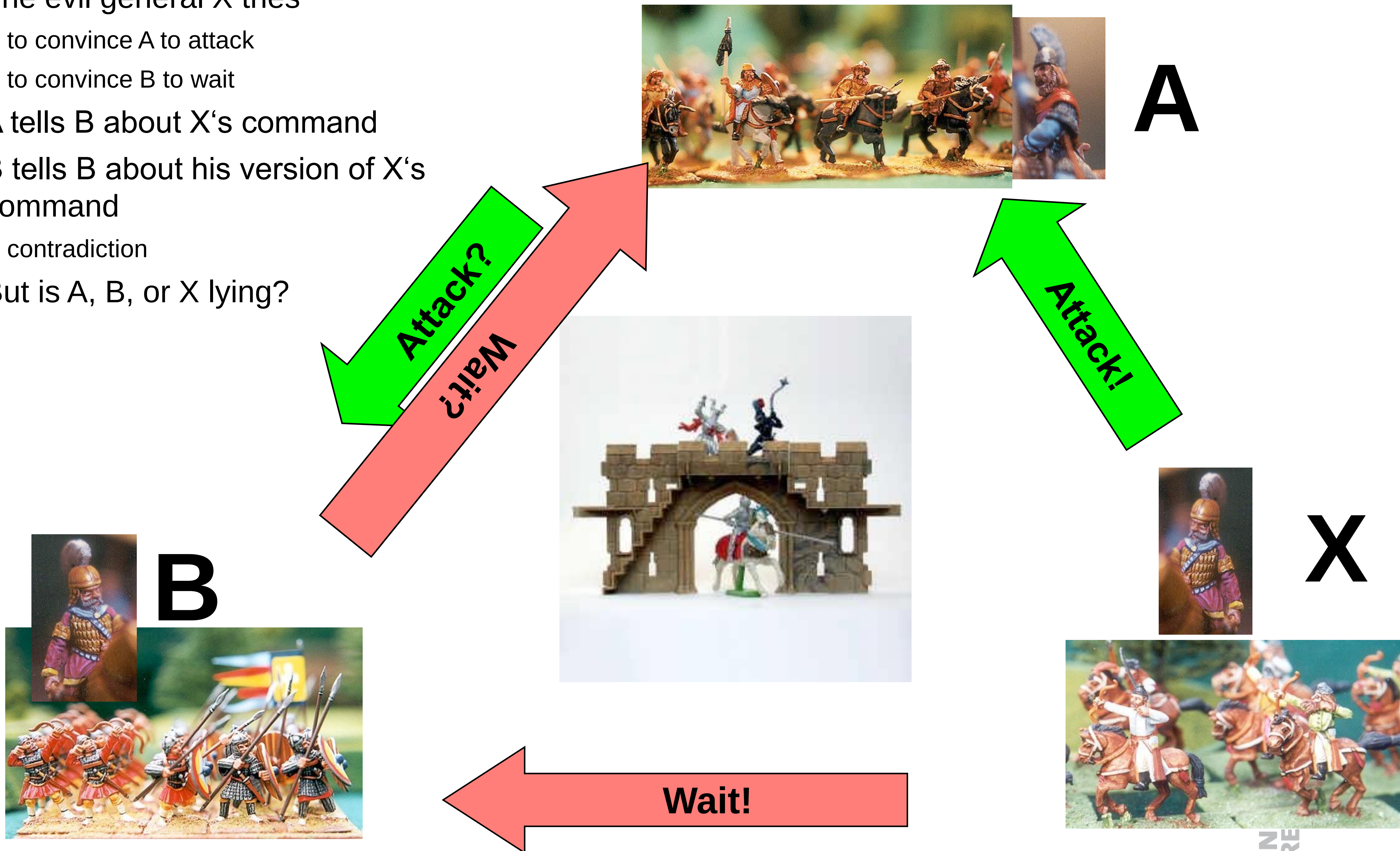


B



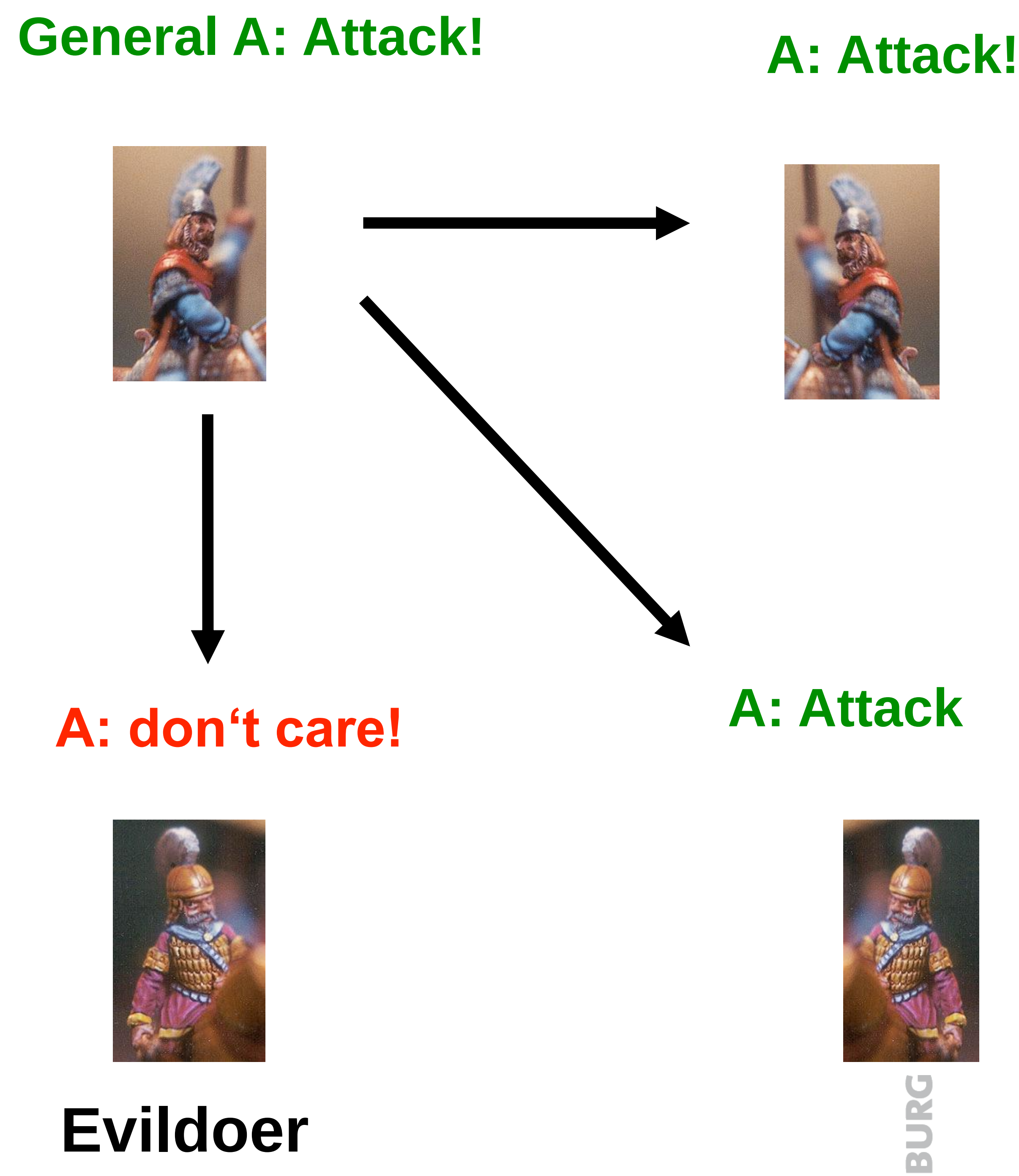
The Problem of Byzantine Generals

- 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?



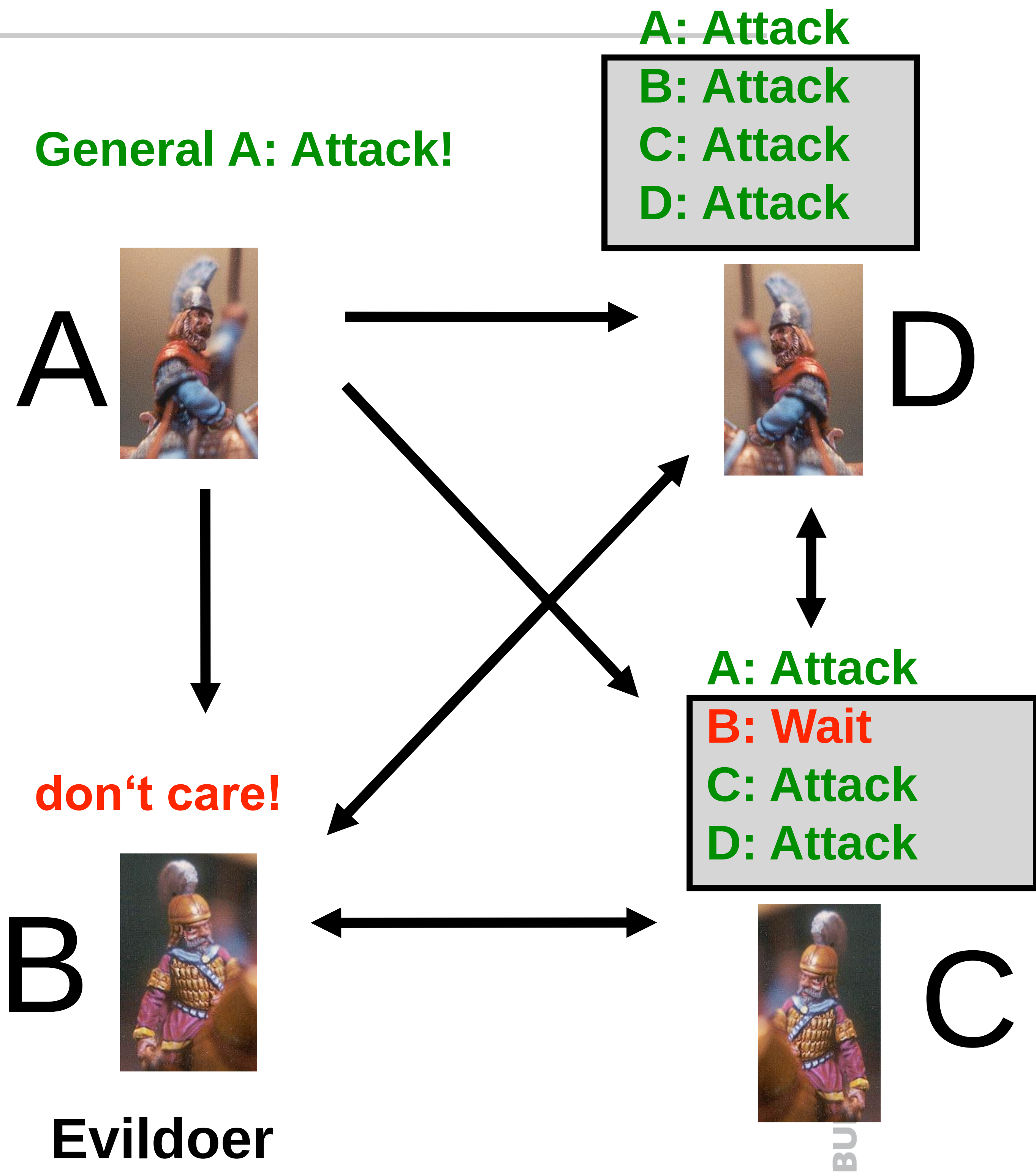
Byzantine Agreement

- 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 commands to all fellow officers
  - What if the general is the renegade?



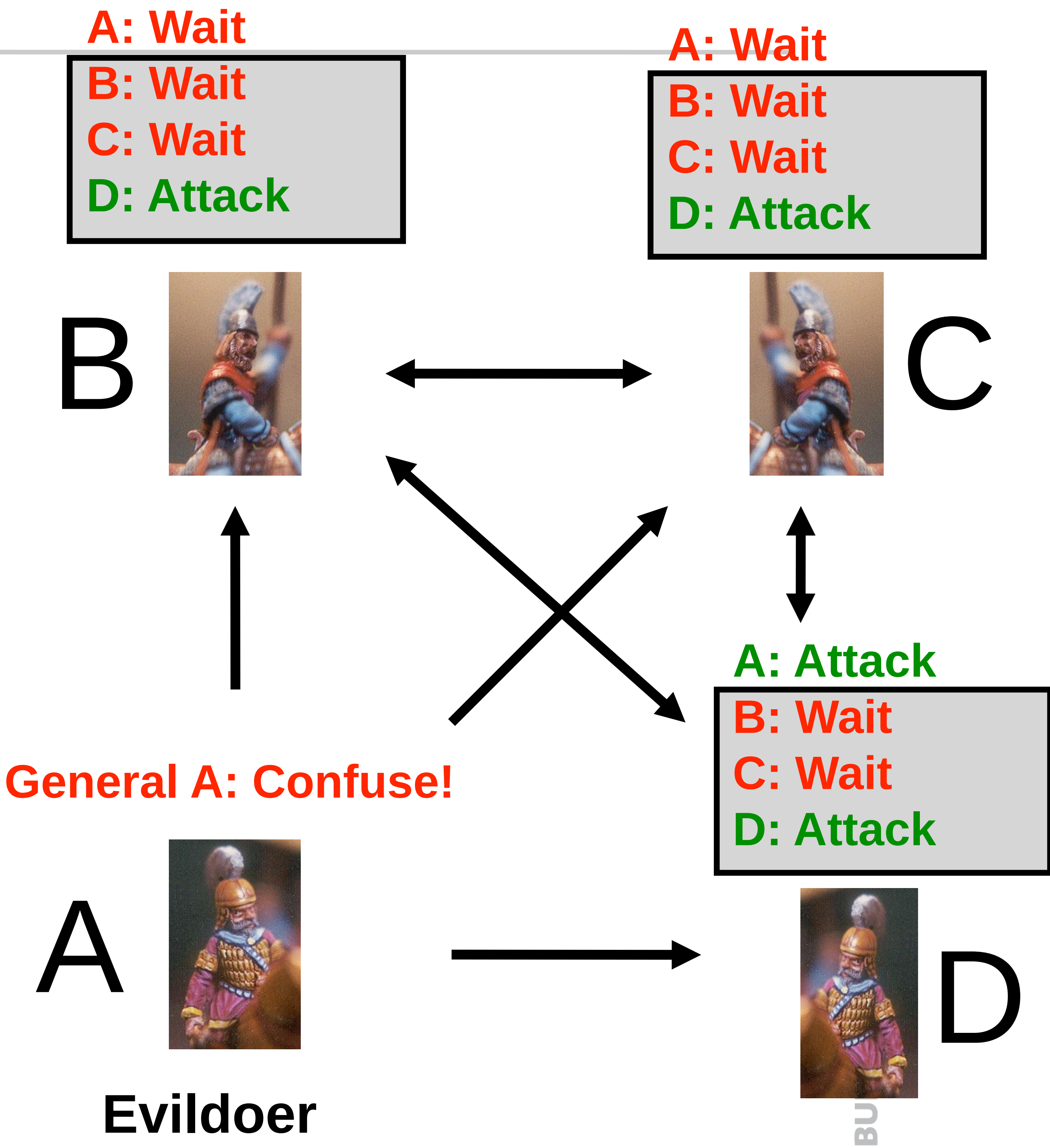
Byzantine Agreement

- 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 commands 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

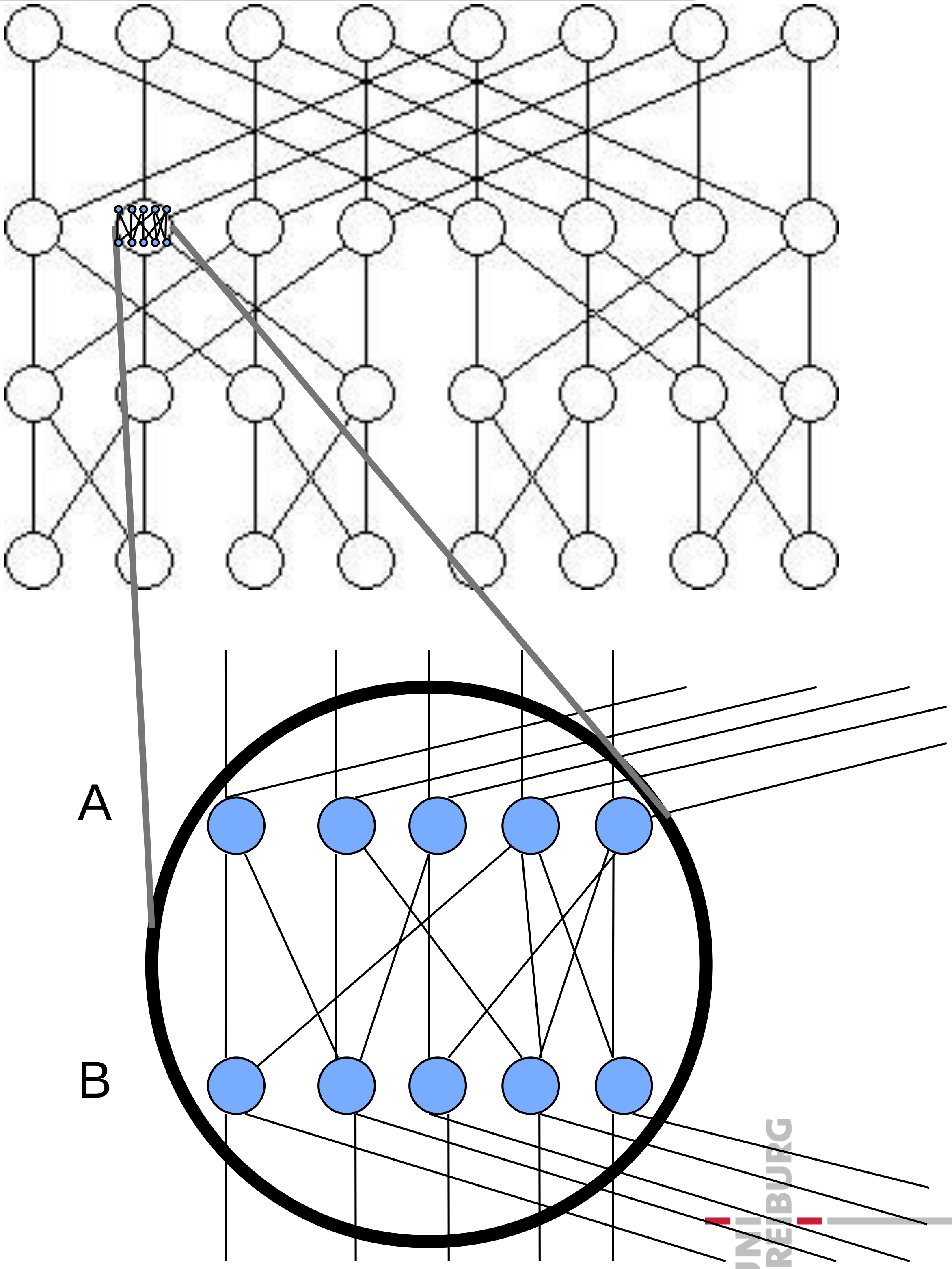
- Theorem
  - If m generals are traitors then 2m+1 generals must be honest to get a Byzantine Agreement
- This bound is sharp if one does not rely on cryptography
- Theorem
  - If a digital signature scheme is working, then an arbitrarily large number of betraying generals can be dealt with
- Solution
  - 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

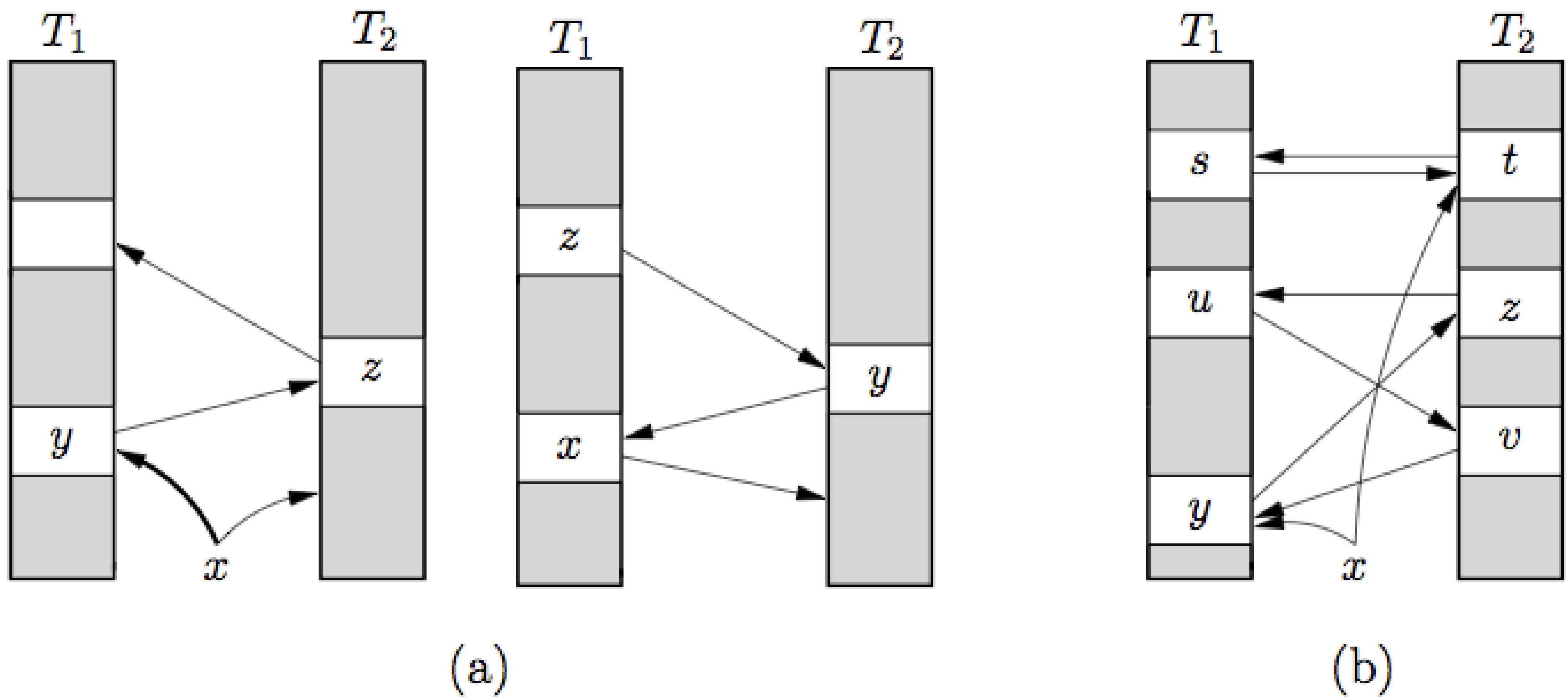


Fig. 1. Examples of CUCKOO HASHING insertion. Arrows show possibilities for moving keys. (a) Key x is successfully inserted by moving keys y and z from one table to the other. (b) Key x cannot be accommodated and a rehash is necessary.

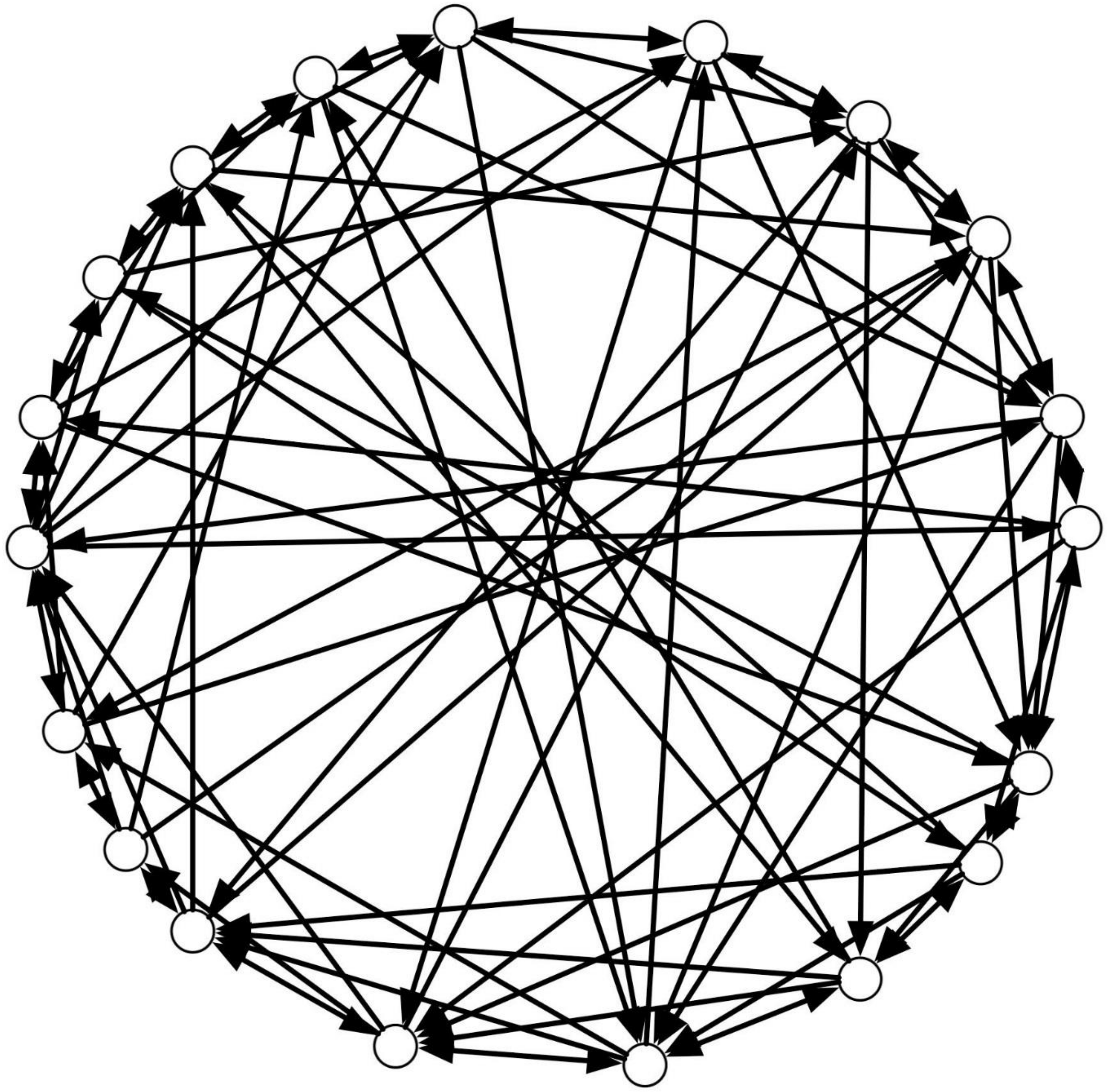
From Cuckoo Hashing
Rasmus Pagh, Flemming Friche Rodler
2004

Efficiency of Cuckoo Hashing

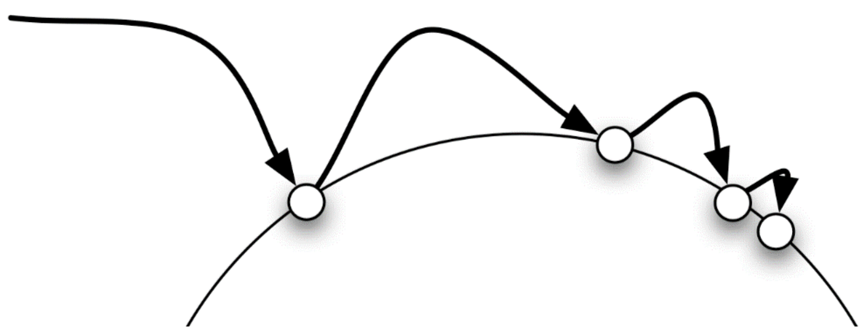
- Theorem
  - Let ϵ>0 then if at most n elements are stored, then Cuckoo Hashing needs a hash space of 2n+ϵ.
- Three hash functions increase the load factor from 1/2 to 91%
- Insert
  - needs O(1) steps in the expectation
  - O(log n) with high probability
- Lookup
  - needs two steps

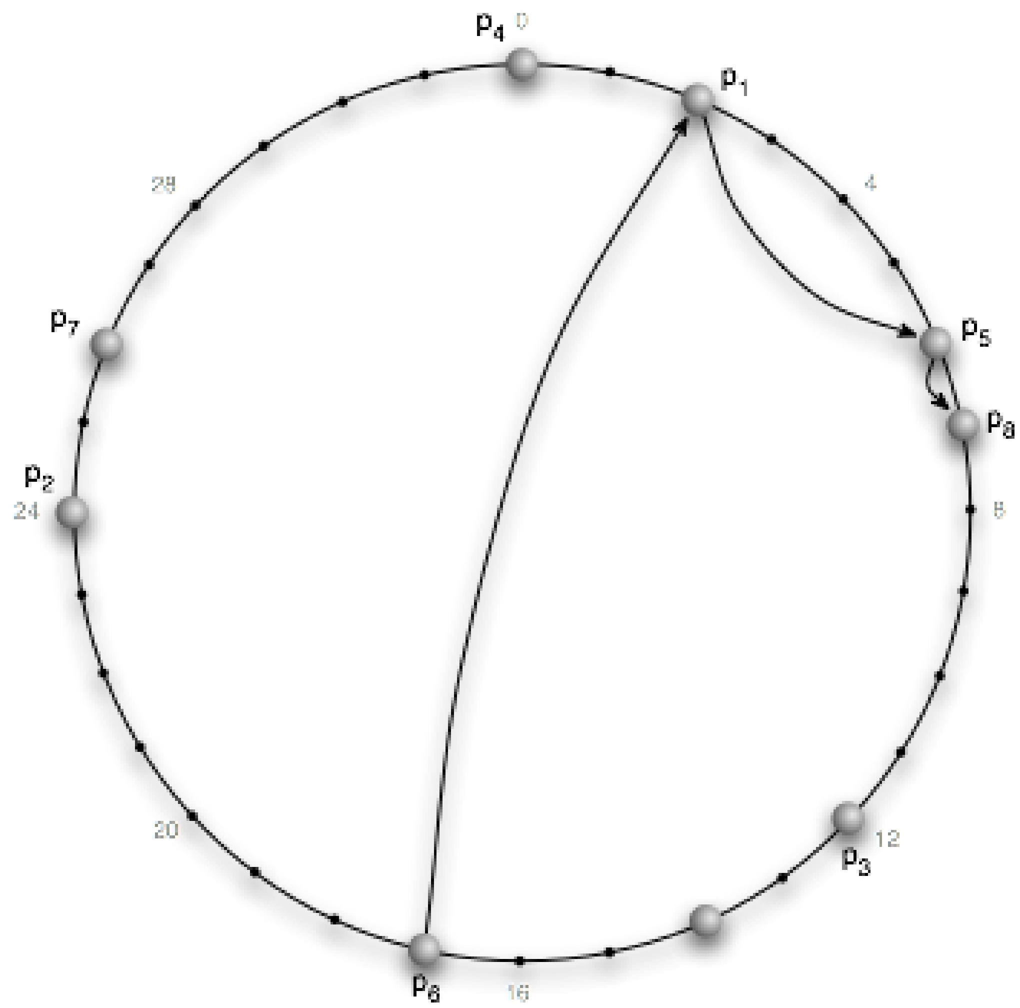
Chord

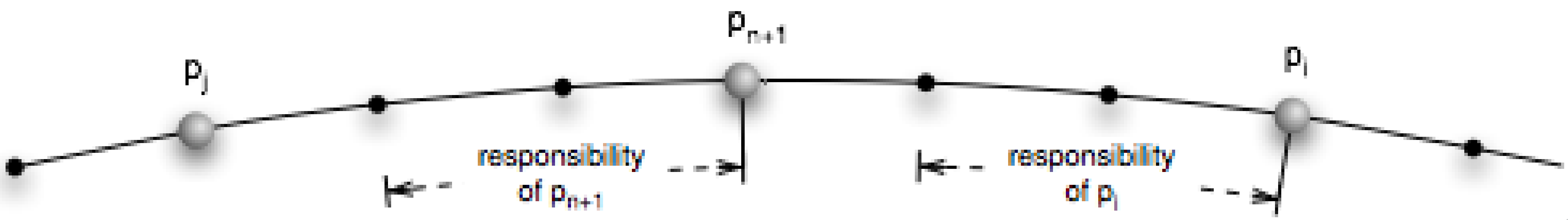
- Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek and Hari Balakrishnan (2001)
- Distributed Hash Table
  - range $\{0,..,2^m-1\}$
  - for sufficient large m
- for this work the range is seen as [0,1)
- Network
  - ring-wise connections
  - shortcuts with exponential increasing distance



Lookup in Chord

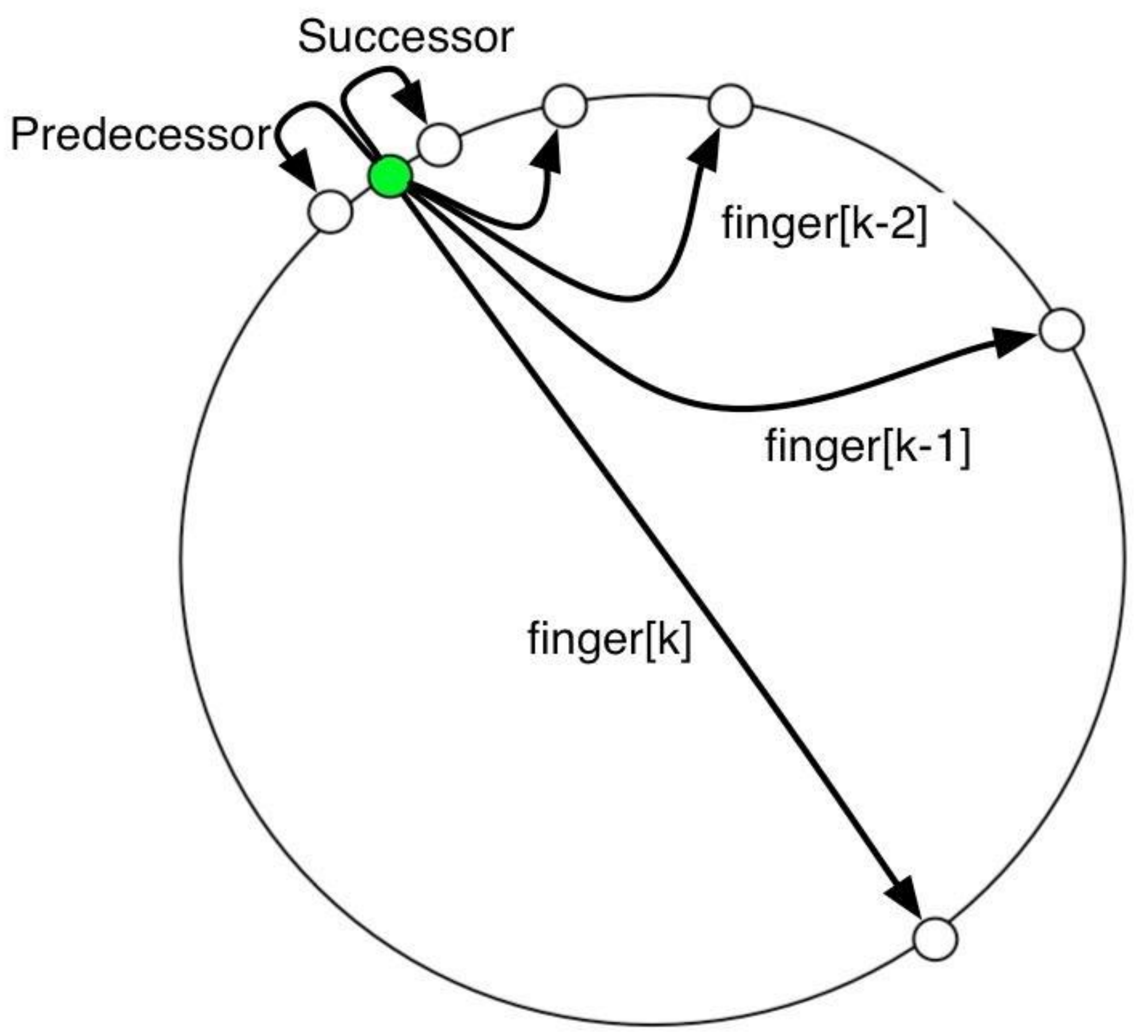






Data Structure of Chord

- For each peer
  - successor link on the ring
  - predecessor link on the ring
  - for all i ∈ {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 $(c \log n)/n$ 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

Peer-to-Peer Networks

14 Security

Christian Ortolf
Technical Faculty
Computer-Networks and Telematics
University of Freiburg