Presentation on:

Tarzan: A Peer-to-Peer Anonymizing Network Layer

Steffen Schott

Computer Networks and Telematics, Freiburg
Prof. Dr. Christian Schindelhauer
Advanced Seminar: Peer-to-Peer Networks
Arne Vater

02/03/2007

Overview

- Motivation
- Architecture and Design
  - Layered Encryption
  - Peer discovery
  - Mimic selection
  - Tunnel setup
  - Tunnel failure and reconstruction
  - Cover traffic
- Security Analysis
- Conclusion

Motivation

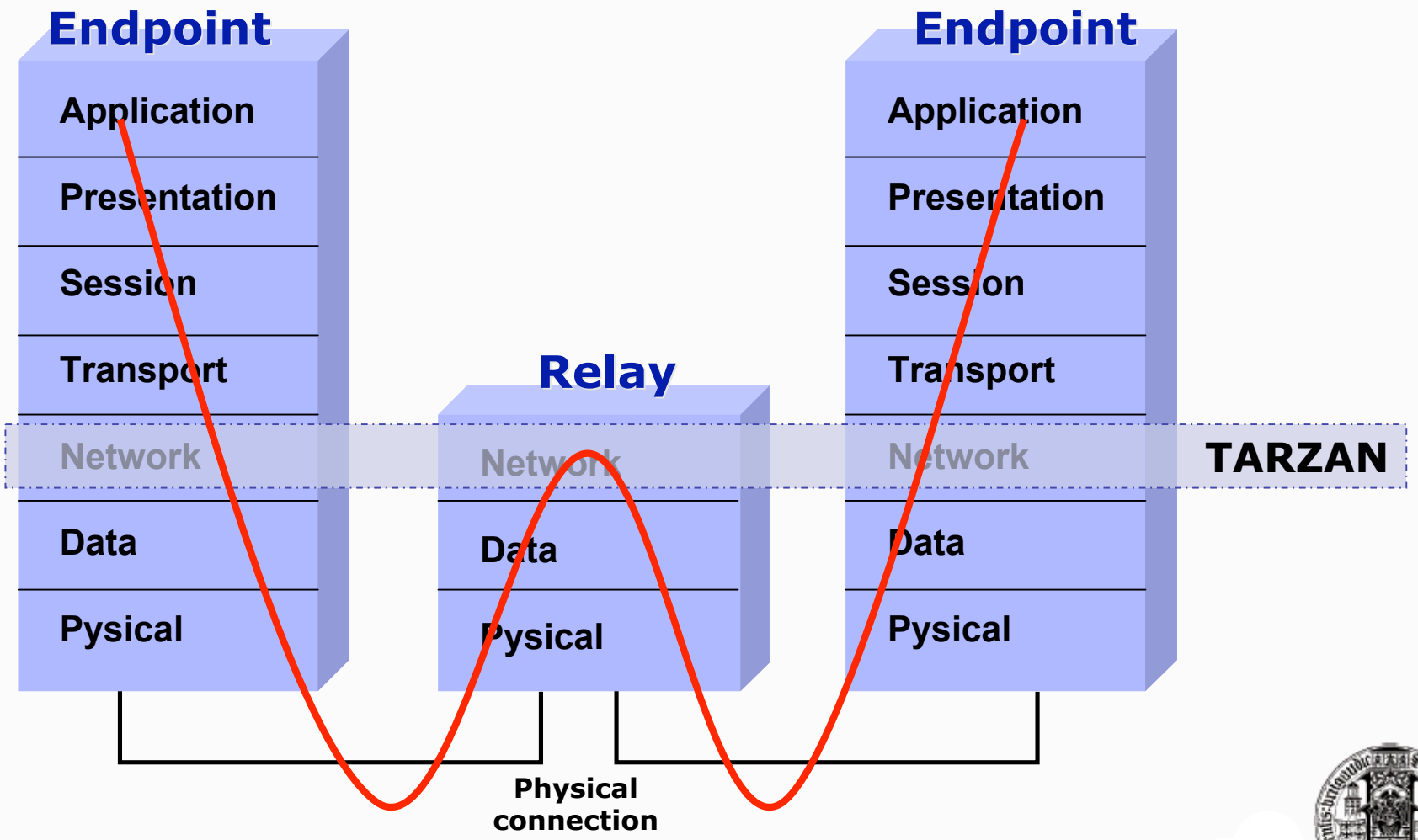
- Tarzan was introduced in 2002 by Michael J. Freedman and Robert Morris
  - Received Paper Award

Idea: Freedman/Morris

- What does Tarzan?

- Provides anonymity to sender or receiver
- Without requiring both to participate
- Peer-to-Peer anonymous network overlay

Motivation

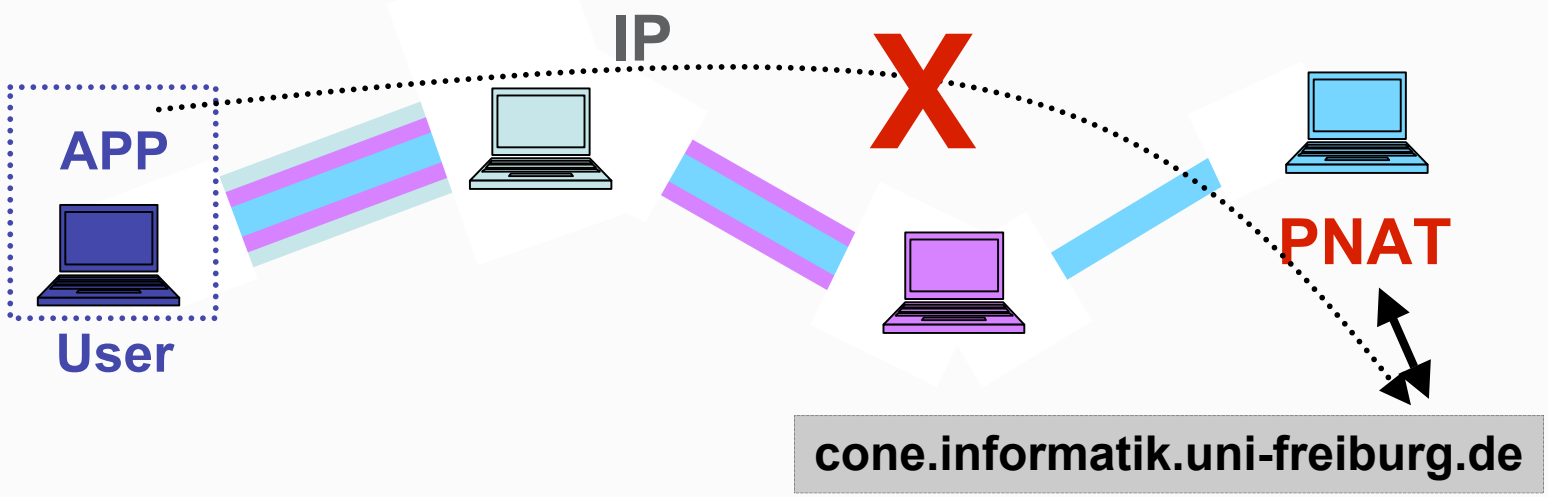

Endpoint
Application
Presentation
Session
Transport
Network
Data
Pysical
Relay
Network
Data
Pysical
Endpoint
Application
Presentation
Session
Transport
Network
Data
Pysical
TARZAN
Physical
connection

Achieving Anonymity

- Techniques used to achieve anonymity:
  - Flexible mixes for tunneling within peers
    - Not like Chaumian Mixes
  - Onion routing style encryption
    - To avoid traceability of path and content disclosure
  - Unforeseen peer selection
    - To protect from adversaries taking over the network by creating specific peers
  - Cover Traffic
    - To lessen traffic analysis attacks
  - Fully Peer-to-Peer
    - No liability at central instance
  - Anonymizing on the IP-Level
    - Independent to applications - no modification needed

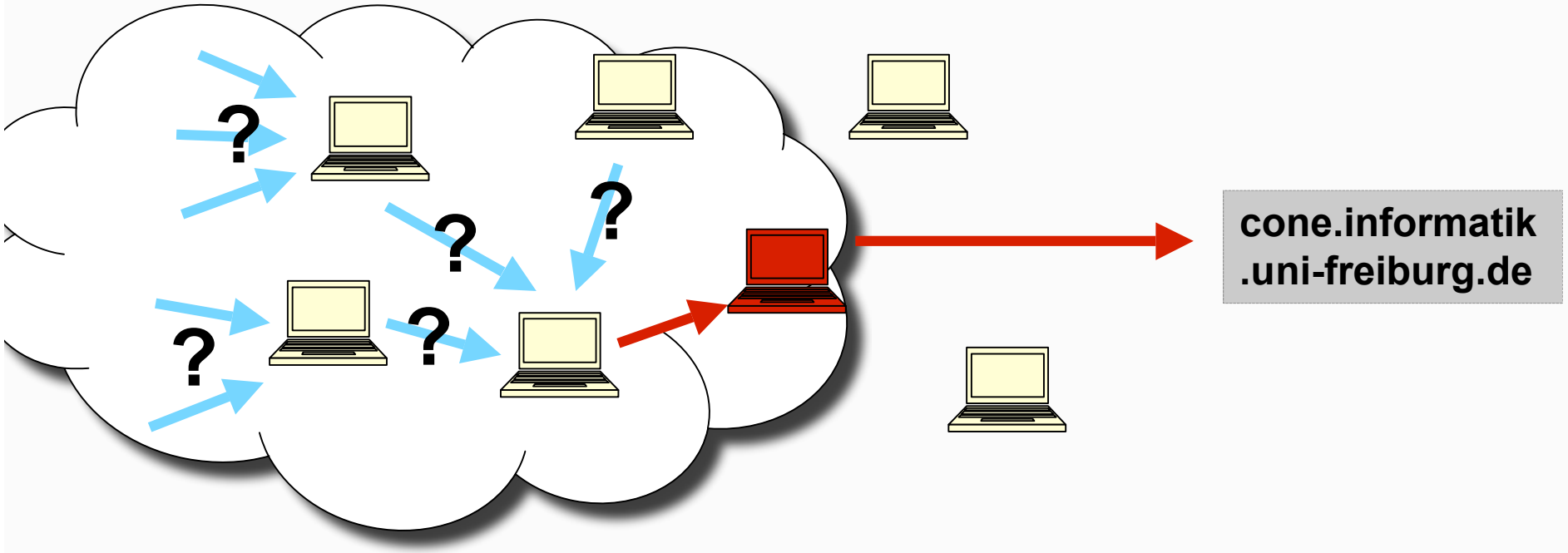
Achieving Anonymity

- Some more general design facts
  - Pseudonymous NAT (PNAT) forwards to servers which are not aware of Tarzan
  - Tunnel initiator sanitizes IP headers, as well as TCP headers if applicable



Source: Freedman/Morris

Achieving Anonymity



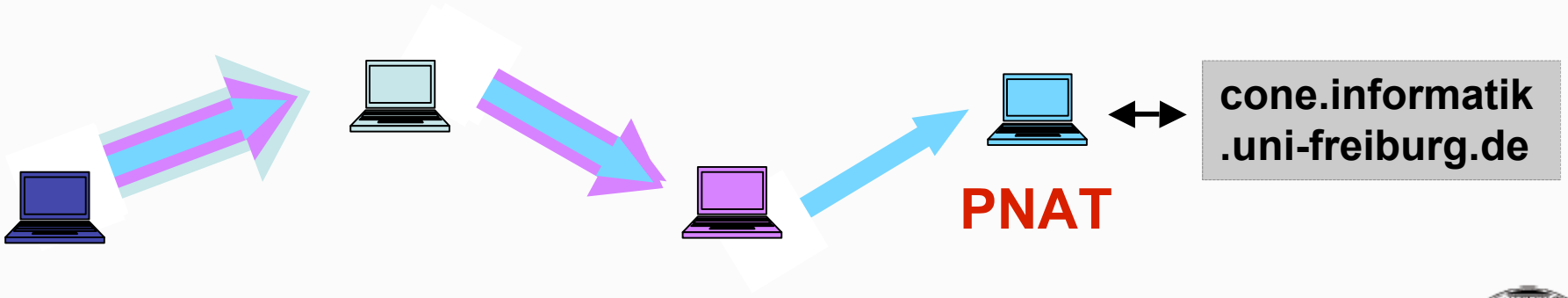
Source: Freedman/Morris

Overview

- Motivation
- Architecture and Design
  - **Layered Encryption**
  - Peer Discovery
  - Mimic Selection
  - Tunnel Setup
  - Tunnel Failure and Reconstruction
  - Cover Traffic
- Security Analysis
- Conclusion

Layered Encryption

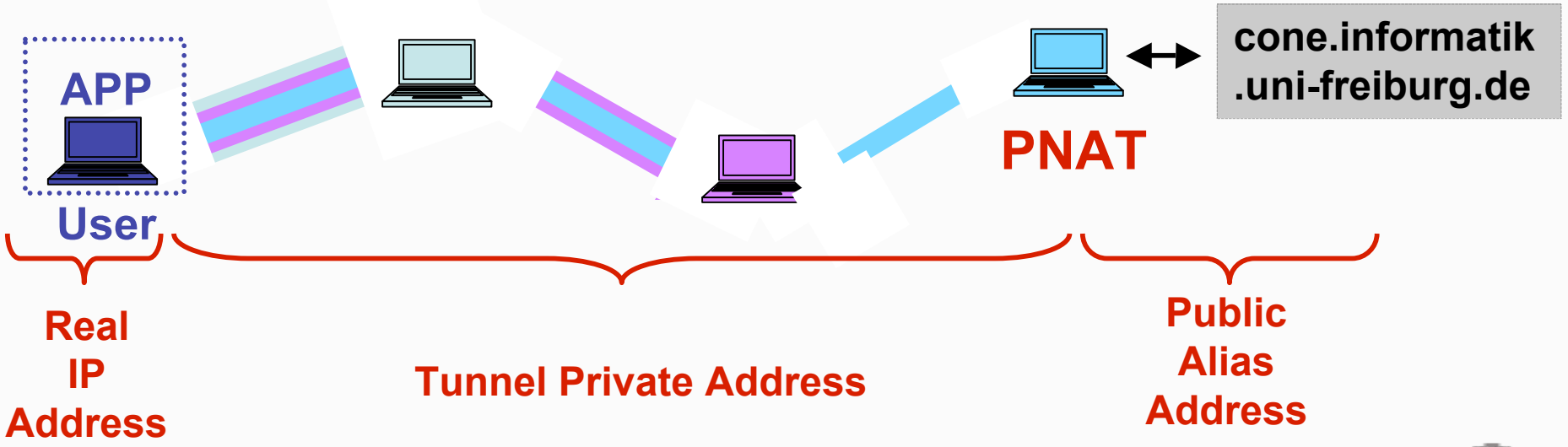
- How do we want to encrypt?
  - Symmetric encryption hides data
  - MAC protects its integrity
  - Separate keys are used in each direction of each relay
  - Therefore, flow tags uniquely identifies each link (of each tunnel)
  - Each leg of the tunnel removes or adds a layer of encryption
    - Like chaumian mixes



Source: Freedman/Morris

Layered Encryption

- Random address assigned
- NATed at beginning and end of the tunnel
- Bulk of the encryption workload on the node seeking anonymity



Source: Freedman/Morris

Encryption Process

- Be
  - $T = (h_1, h_2, \ldots, h_1, h_{pnat})$ Tunnel → short version: $T = (h_1, h_2, h_{pnat})$
  - B_i = block to receive by node i
  - ENC = encryption
  - MAC = fingerprint
  - seq = sequence number

- General Rule for each node:

$$c_i = ENC(ek_{h_i}, \{B_{i+1}\})$$
$$a_i = MAC(ik_{h_i}, \{seq, c_i\})$$
$$B_i = \{seq, c_i, a_i\}$$

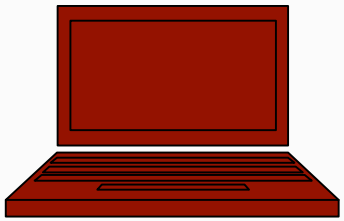
Example for T_s

$$c_{pnat} = ENC(ek_{h_{pnat}}, \{B_{pnat+1}\})$$
$$a_{pnat} = MAC(ik_{h_{pnat}}, \{seq, c_{pnat}\})$$
$$B_{pnat} = \{seq, c_{pnat}, a_{pnat}\}$$
$$c_2 = ENC(ek_{h_2}, \{B_{pnat}\})$$
$$a_2 = MAC(ik_{h_2}, \{seq, c_2\})$$
$$B_2 = \{seq, c_2, a_2\}$$
$$c_1 = ENC(ek_{h_1}, \{B_2\})$$
$$a_1 = MAC(ik_{h_1}, \{seq, c_1\})$$
$$B_1 = \{seq, c_1, a_1\}$$

?

- Every tunnel has an end…

  Any consequences?



PNAT

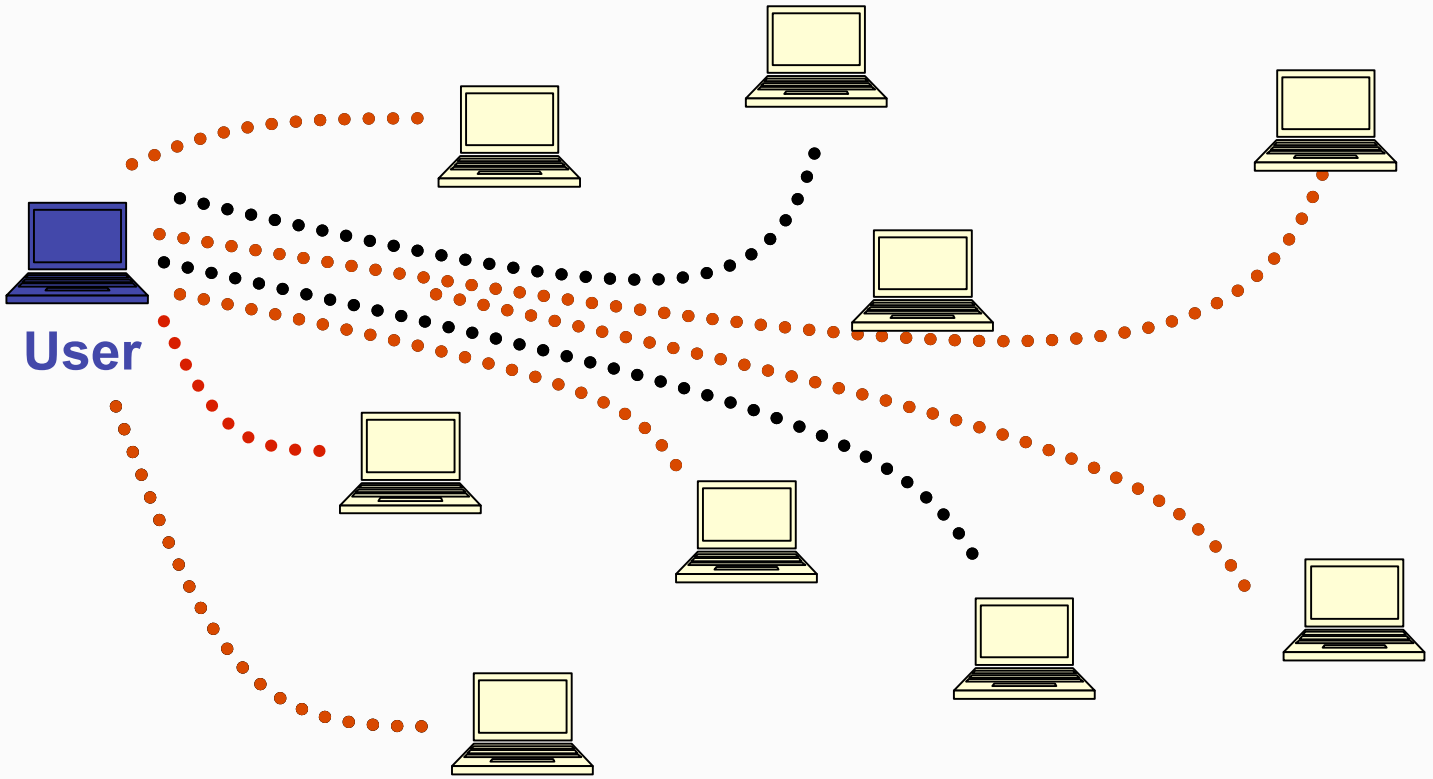
Overview

- Motivation
- Architecture and Design
  - Layered Encryption
  - **Peer Discovery**
  - Mimic Selection
  - Tunnel Setup
  - Tunnel Failure and Reconstruction
  - Cover Traffic
- Security Analysis
- Conclusion

Peer Discovery

- Objective: Assigning neighbors - in a decentralized but verifiable manner
  - Each node generates its public key locally the first time it enters the network
  - Knowing initially only a few nodes
  - Peer discovery by simple gossip-based protocol
    - By sending `{ipaddr, port, hash(pubkey)}` - tuples
  - Goal: to learn about all network resources - fully connected

Peer Discovery



Source: Freedman/Morris

Protocol

- Protocol supports: initialization, redirection and maintenance

  - Initialization: transfer entire neighbor list - from randomly contacted neighbor
  - Redirection: redirecting new nodes to random neighbor (to shed load)
  - Maintenance: provide only new information to a node's database
    - Differences calculated efficiently by performing k-ary searches on prefix-aggregated hashes of the set elements
      - $H_{[n]} \rightarrow H_{[n]/k} \rightarrow H_{[n]/k^2} \rightarrow O(\log_k n)$

  - Hash values of node a's sorted set V_a – approx. $(k-1)$ values sent at a time

$$H_i = hash(\ldots hash(hash(V_a[1]) + V_a[2]) \ldots + V_a[i])$$

IP-Tables

- Building IP-Tables:
  - Differentiation: unvalidated (U_a) and validated addresses (V_a) of node ***a***
  - Only V_a in IP-Table → for mimic & tunnel selection
  - Validation by discovery request
  - Stops an adversary from injecting arbitrary tuples into a peer database
  - Contacting neighbors in U_a before retrying neighbors in V_a
  - Prunes inactive neighbors
  - Learns and validates in $O(n)$ connections

?

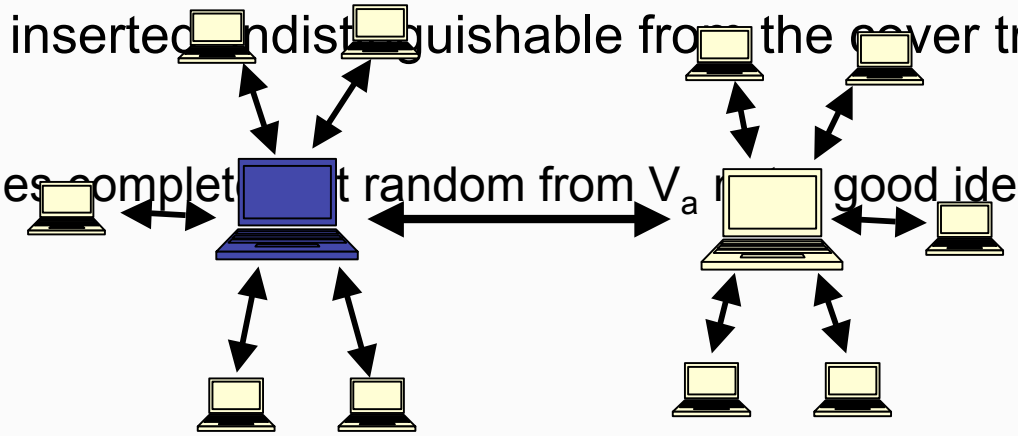
- What is probably the most negative fact about this algorithm?

Overview

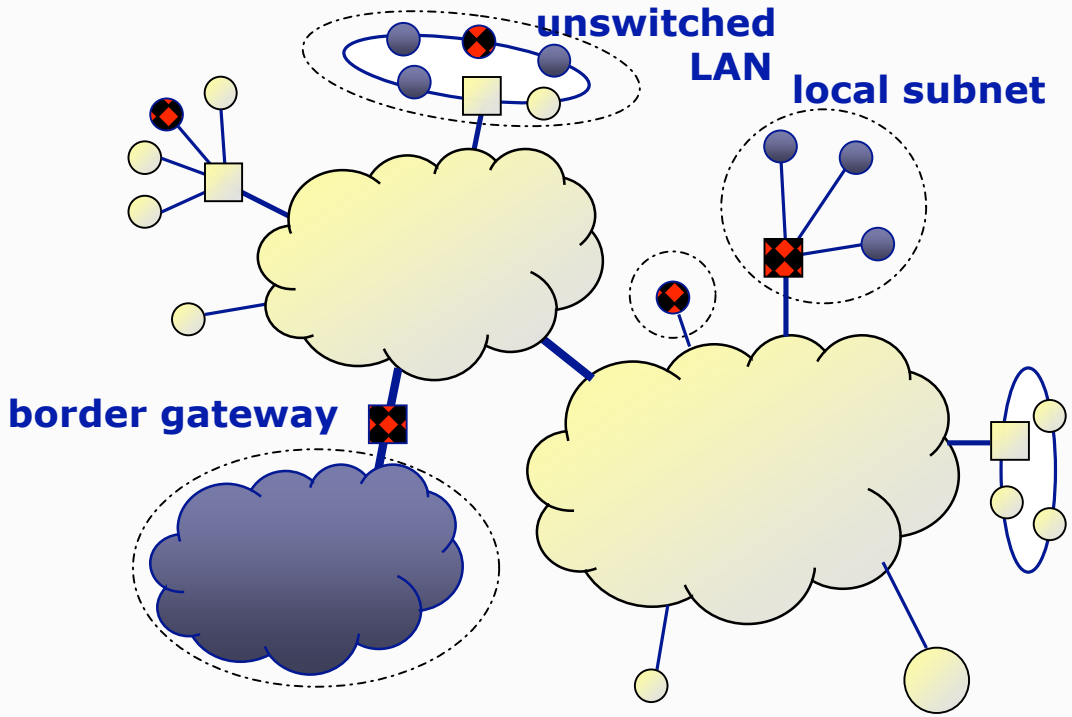
- Motivation
- Architecture and Design
  - Layered Encryption
  - Peer Discovery
  - **Mimic Selection**
  - Tunnel Setup
  - Tunnel Failure and Reconstruction
  - Cover Traffic
- Security Analysis
- Conclusion

Mimic Selection

- Threat: wide-spread eavesdropper can analyse traffic patterns
- Finding partners for cover traffic:
  - Every node upon joining asks `k` nodes to exchange dummy/mimic traffic
  - An expected `k` nodes select this node as they look for their own mimics
  - Goal: establishes a bidirectional, time-invariant packet stream with all `E[K]=2k` mimic nodes
  - After successfully discovery - symmetric key for encryption is exchanged for link encoding
- Now, real data can be inserted indistinguishable from the cover traffic
- Can be anyone?
  - Simply choosing nodes completely random from V_a not good idea



Threats



- honest node
- malicious node
- spoofed node
- honest router
- malicious router
- corrupted domain

Idea: Freedman/Morris

Hashing

- Thus
  - Tarzan uses three-level hierarchy chord ring (DHT)
  - First chooses from /16 subnets, then /24 and finally from the rest
  - Node a's i^{th} mimic =: M_{a-i}

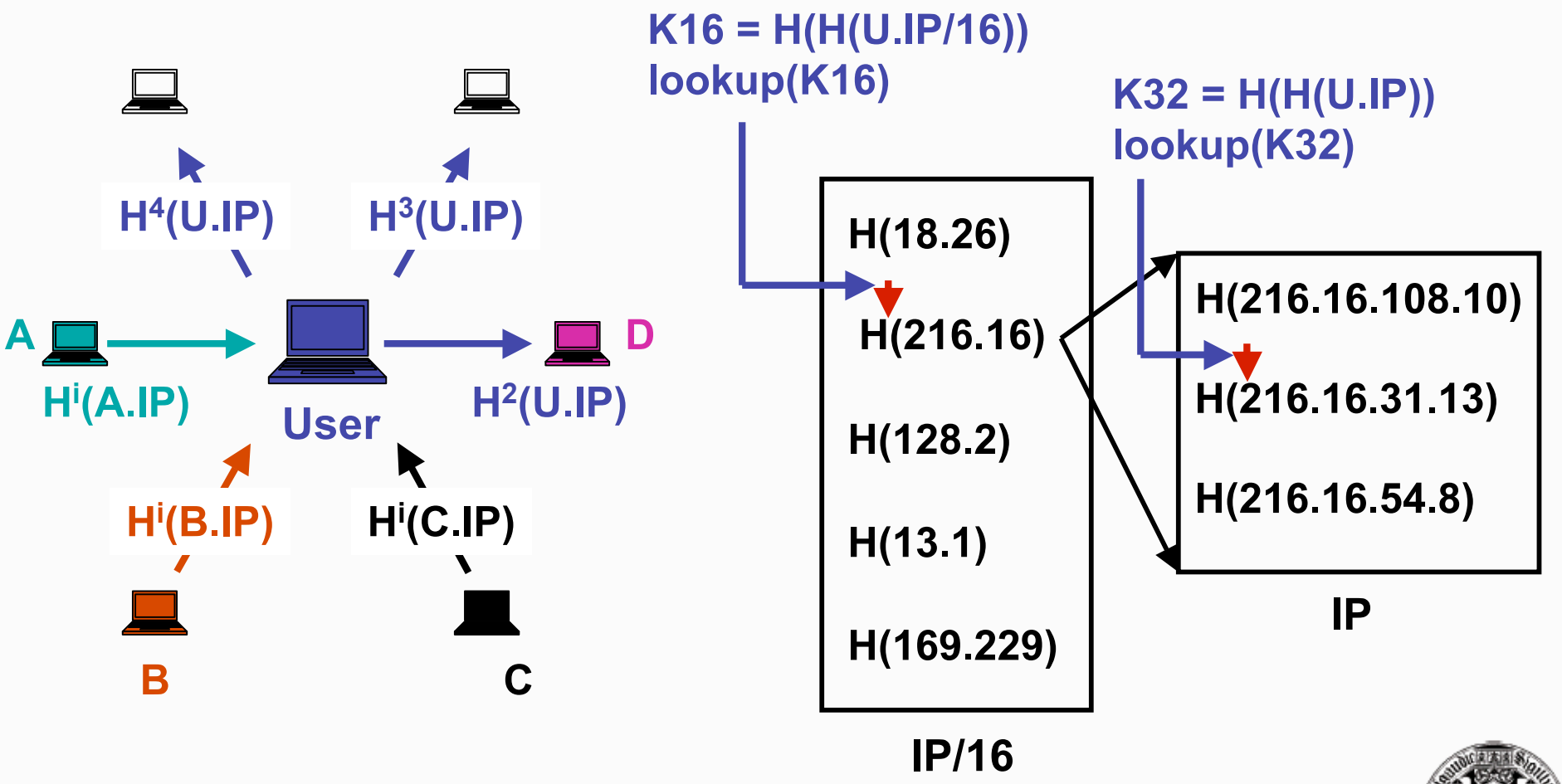
    where M_{a-i} is the smallest $id \geq id^i = lookup^i(a.ipaddr)$

    and $lookup_d(a.ipaddr) = hash(a.ipaddr/d,date)$
  - So:

    $lookup^i_d(a.ipaddr) = hash(..hash(hash(a.ipaddr/d,date))..)$

    with d element {/16, /24, /32}

Hashing



Source: Freedman/Morris

Connecting a Mimic

- Steps:
  - Node `a` sends mimic request to M_{a-i} including `{a.ipaddr, i}`
  - M_{a-i} =: `b` only accepts mimic establishment if:
    1. $1 < i \leq (k+1)$
    2. $b.lookup^{i}(a.ipaddr) = b$

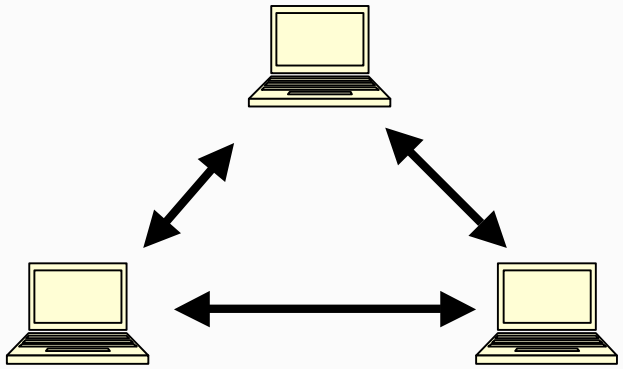
    to verify that `b` is true i-th mimic of `a`
  - If lookup-check fails:

    1st case: `a` and `b` have different network view

    2nd case: `a` already contacted `c`, but `c` didn't respond

?

- If A and B are mimics. How probable is it, them to have a common second mimic?

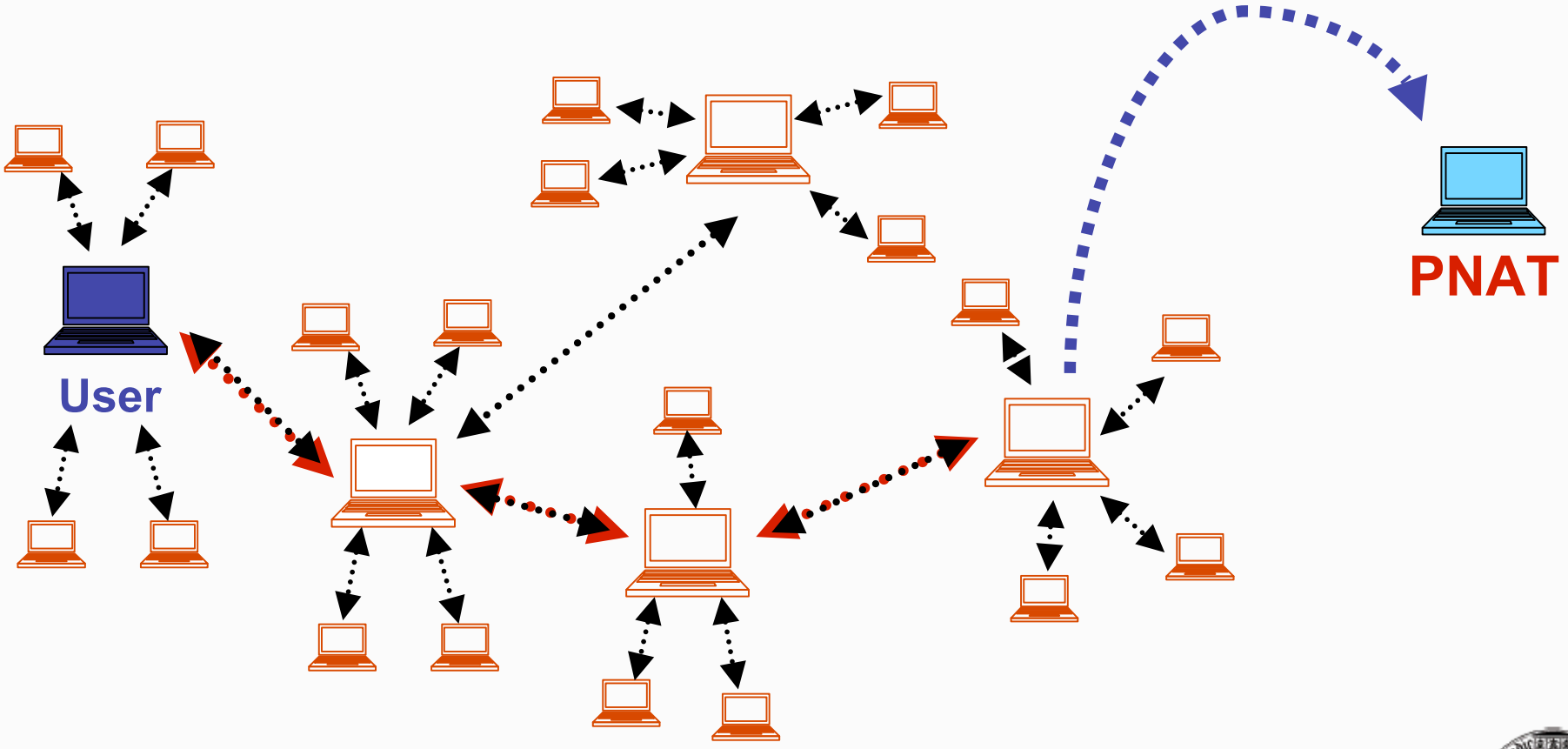


Overview

- Motivation
- Architecture and Design
  - Layered Encryption
  - Peer Discovery
  - Mimic Selection
  - **Tunnel Setup**
  - **Tunnel Failure and Reconstruction**
  - Cover Traffic
- Security Analysis
- Conclusion

Tunnel Setup

- Selecting tunnel nodes



Idea: Freedman/Morris

1.	{fromIP,flowID}	{integrityKey,toIP,flowID,SymKey}
2.	{fromIP,flowID}	{revIntegrityKey,toIP,flowID,reverseSymKey}
..		

- `O(length)` public-key operations and and $O(length^2)$ inter-relay messages to complete
- Overhead
  - tunnel setup: approx. `20ms/hop`
  - for packet forwarding: approx. `1ms/hop` (each)

Tunnel Failure and Reconstruction

- Initiator regularly sends ping messages to the PNAT
  - Upon multiple unsuccessful pings to PNAT - then pings to each relay

1st case: PNAT unreachable, h_1 responds

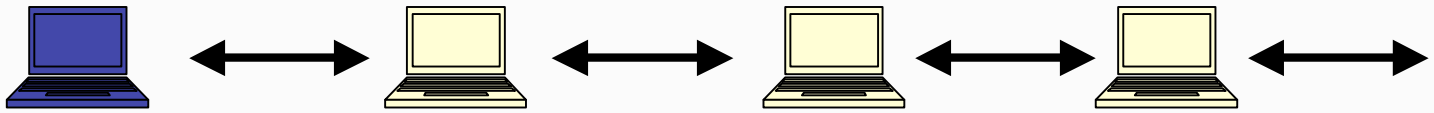
- New PNAT will be chosen randomly

2nd case: any relay < h_1 doesn't respond

- Tunnel is partially reconstructed - PNAT stays the same
- So that higher level connections, such as TCP, do not die upon tunnel failure
- Example: h_{i+1} doesn't respond - rebuild the tunnel from h_i forward
  - $T' = (h_1, \ldots, h_i, h_{i+1}', \ldots, h_l', h_{pnat})$
- Upon multiple unsuccessful attempts, the initiator decrements i by one and reattempts reconstruction

?

- What if one relay simply doesn‘t forward traffic?



Overview

- Motivation
- Architecture and Design
  - Layered Encryption
  - Peer Discovery
  - Mimic Selection
  - Tunnel Setup
  - Tunnel Failure and Reconstruction
  - **Cover Traffic**
- Security Analysis
- Conclusion

Cover Traffic – Unifying Traffic Patterns

- Mimics links are symmetrically encrypted on top of the tunnel → cover traffic indistinguishable from data flows
- Incoming cover traffic can be dropped on demand or rebalanced on any outgoing links
- No congestion control or retransmission in relays
- Freedman and Morris are giving two equations

Equations

- Outgoing DATA rate to single tunnel ≤ ⅓ Total incoming rate (data + cover)
  - node cannot be identified as being a clear source of data
- ⅓ Total incoming rate (data + cover) ≤ Total Outgoing rate (data + cover) (=upper bound)
  - Always have some cover traffic for adjustments
  - Provide anonymity to its neighbors
  - Stops node from being clear sink of traffic

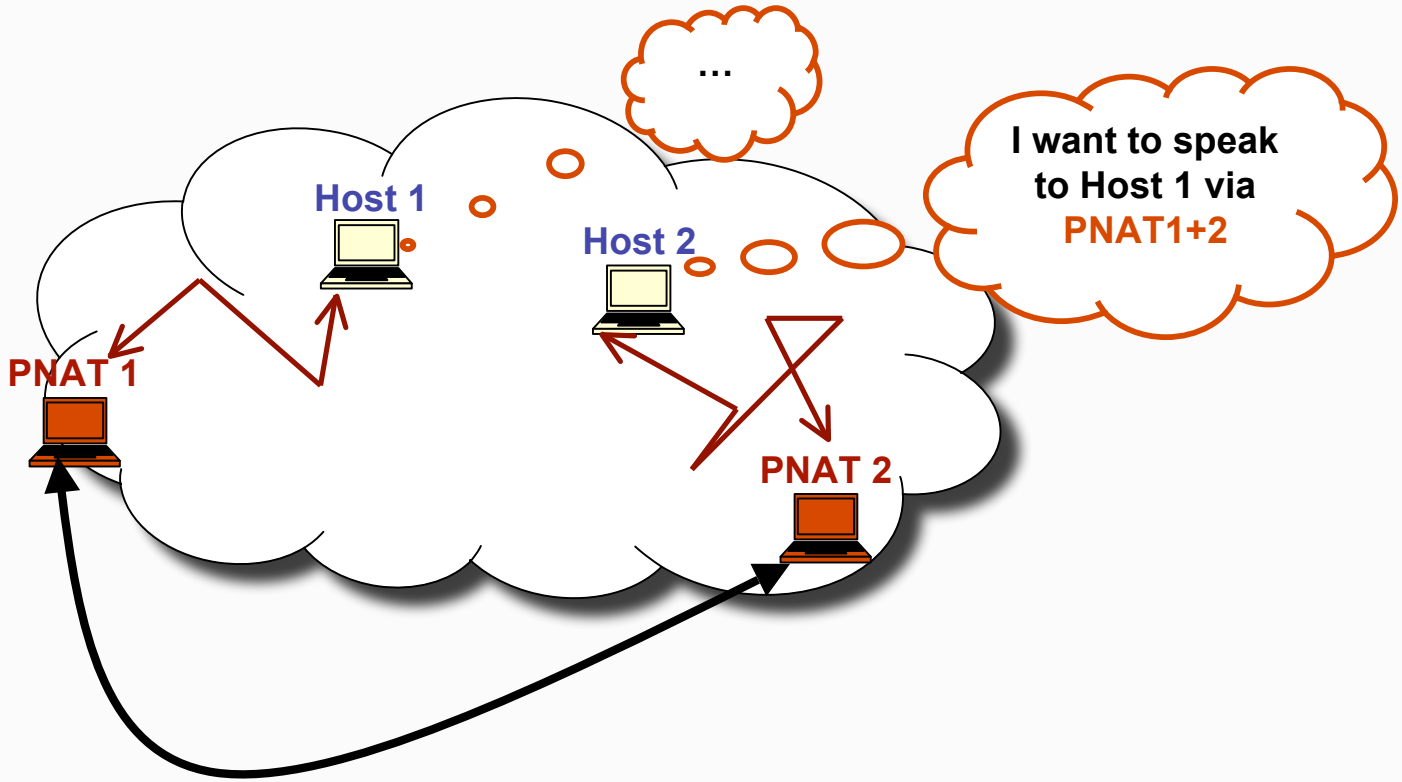
and

Total Outgoing rate (data + cover) ≤ Maximum total incoming rate + ε (=lower bound)

- Again: node cannot be identified as being a clear source of data
- ε - to cooperatively raise their maximum traffic levels

Further Possibilities

- Achieving both sender and recipient anonymity

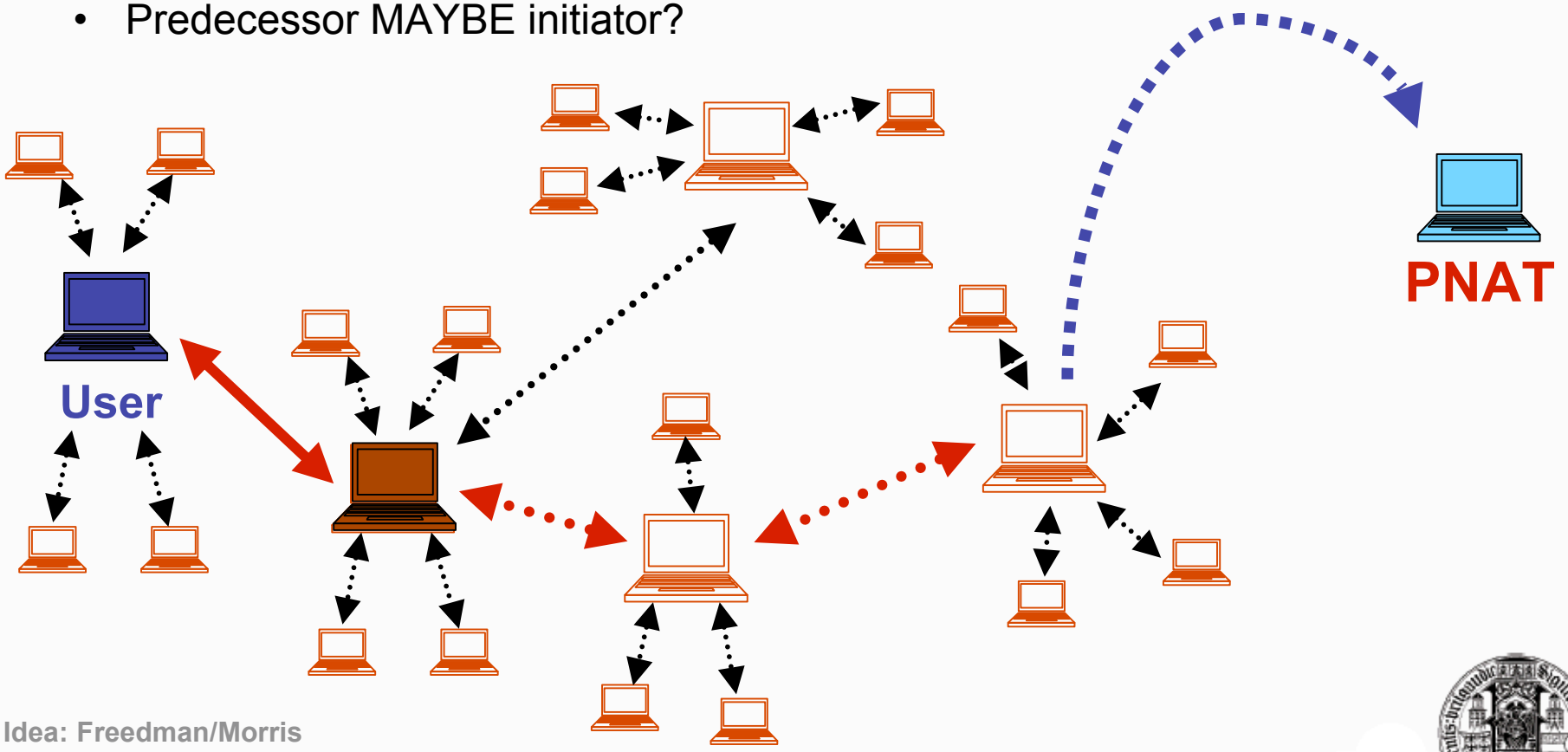


Overview

- Motivation
- Architecture and Design
  - …
  - …
- Security Analysis
  - **Prevented Attacks**
  - **Possible Attacks**
  - **Possible Improvements**
- Conclusion

Security Analysis

- Who knows his own role?
  - Node h_1 to h_{1-1} just know that relay, but not position
  - Predecessor MAYBE initiator?



Idea: Freedman/Morris

Prevented Attacks

- Various attack given in open-admission, self-organized peer-to-peer models have been faced!
  - Attacks through corrupt gossiping
    - Only if all initially known peers are malicious will keep wrong IP-Table
  - Attacks given by open admission
    - Adversary might control many peers in some domains but not the Tarzan network, thanks to subnet-hierarchy hashes for IP-Tables
    - Public keys are gossiped and not distributed directly
  - Attacks per ignoring neighbor-selection algorithm
    - Mimics cannot be „generated“ due to hash algorithm
    - On tunnel setup, mimics of all relay are verified
  - Attacks by adaptive, compromising adversary
    - Tunnel duration and mimic stability probably to small for adversary
    - Situation far more difficult for adversary than in a central core network

Prevented Attacks

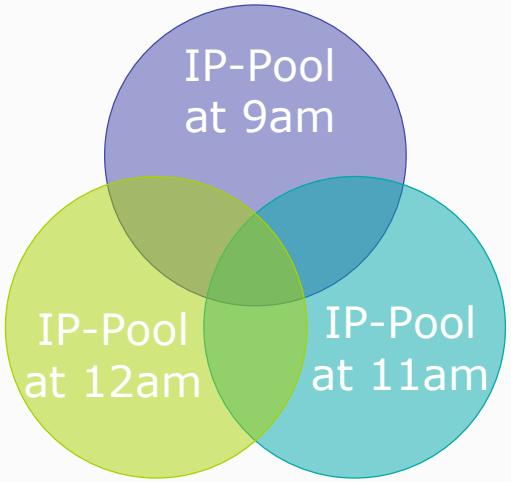
- Further attacks …
  - Attacks of mimic nodes by sudden mutual omission of cover traffic
    - Should not be successful due to traffic invariants
  - Attacks by interpreting content
    - Should be impossible due to complex encryption and integrity mechanisms
    - Except at PNAT
  - Attacks through traffic analysis
    - Weak possibilities, and only for relays
  - Attacks, that take advantage from modifying packets (except omission)
    - Probably will be dropped caused by integrity checks

Possible Attacks

- Attack on tunnel reconstruction protocol
  - Simply not forward traffic for two corresponding flow identifiers by h_i
  - The initiator will suspect h_{i+1} not to work and will be trying another mimic of h_i
  - h_i can repeat that until h_{i+1} is an adversary mimic as well, and so on for h_{i+1}
  - Attack can be avoided if reconstruction starts at node h_{i-1}
  - So far not part of the Tarzan design

Intersection Attack - Passive Logging Attack

- Most powerful, while extremely easy to fulfill
- Few means of defending
- Only single peer in the system is needed to obtain full IP-Table
- Taking a collection of timely disjoint set of nodes - which contain the initiator
- Just intersecting those sets will decrease list of possible IPs
- Even extremely efficient for low bandwidth protocols like SMTP



Other Possible Attacks

- A capable adversary might see a request from PNAT to some webserver
  + sees the forwarding to h_1
  - This is as h_{pnat} an h_1 are no mimics - no cover traffic is exchanged
  - Few was said in Paper about batching of data packets et al. is applied to avoid linkability of h_{pnat} to h_1
  - Batching in 20msec intervals only, done by every relay
- Traffic analysis by relay limited yet possible
  - Counting packets + measurement of response times
  - Estimation of distance from initiator
    - Example: Maximum of 3 hops – Just expected 5 x 6 + 1 possible initiators
- Further traffic analysis
  - If a global eavesdropper has various malicious peers in tunnels, which one by one stop forwarding traffic for short time
  - Global eavesdropper can notice stop of traffic from webserver to PNAT

Other Possible Attacks

- Attacks by sending data via suspicious node (possible initiator)
  - Estimating outgoing data rate ≤ ⅓ total incoming rate (data + traffic)
  - Set up tunnel via suspicious node + send data
  - If node rejects tunnel setup or not the full amount of data passes, probable relay or initiator of real data
  - Attackers might exceed own upper bound of outgoing DATA (⅓ of total Incoming)

Possible Improvements

- Setup of various tunnels at a time to same or even different PNAT
  - Gaining connection reliability
  - Can make timing/traffic analysis harder (even for relay peers)
- Slight variation of tunnel reconstruction protocol to avoid interference of adversary
  → Rebuild tunnel from h_{i-1} if h_{i+1} doesn't respond
- Further batching of packets at PNAT
  - To lessen possibility of traffic analysis
- Using a proxy to lessen risk of intersection attack

Overview

- Motivation
- Architecture and Design
  - …
  - …
- Security Analysis
  - Prevented Attacks
  - Possible Attacks
  - Possible Improvements
- **Conclusion**

Conclusion

- Fully P2P anonymizing network layer
- Independent to applications
- Protecting against various attacks of edge analysis
- Efficiently constructed – up to real-time
- But: Some known passive logging attacks

Any Questions?

Any Questions?

- Introducing Tarzan …

Source: Harold F. Schiffman haroldfs@ccat.sas.upenn.edu

Some Literatur

☎Michael J. Freedman and Robert Morris *Tarzan: A Peer-to-Peer Anonymizing Network Layer*, in Proceedings of the 9th ACM Conference on Computer and Communications Security, Washington, D.C., 2002

And slides: http://www.scs.stanford.edu/mfreed/docs/tarzan-ccs02-slides.pdf

☎M. Wright and M. Adler and B. Levine and C. Shields, *Defending anonymous communication against passive logging attacks*, in Proc. IEEE Symposium on Research in Security and Privacy, Berkeley, CA, May 2003

☎Andrei Serjantov and Peter Sewell, *Passive Attack Analysis for Connection-Based Anonymity Systems*, University of Cambridge, 2003

☎Alan Mislove Gaurav, *AP3: Cooperative, decentralized anonymous communication*, in Proceedings of the 11th workshop on ACM SIGOPS European workshop: beyond the PC, Leuven, Belgium, 2004

☎JAP Anon Proxy, http://anon.inf.tu-dresden.de/