

PeerMint: Decentralized and Secure Accounting for Peer-to-Peer Applications

David Hausheer

*Computer Engineering and Networks Laboratory, TIK
Swiss Federal Institute of Technology, ETH Zürich
E-Mail: hausheer@tik.ee.ethz.ch*



Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich



Outline

- Motivation and Objective
- Goals and Design Space of Accounting
- Distributed Redundant Accounting
- PeerMint Design and Implementation
- Analytical and Experimental Results
- Conclusions and Future Work



Motivation and Objective

□ Benefits of P2P systems:

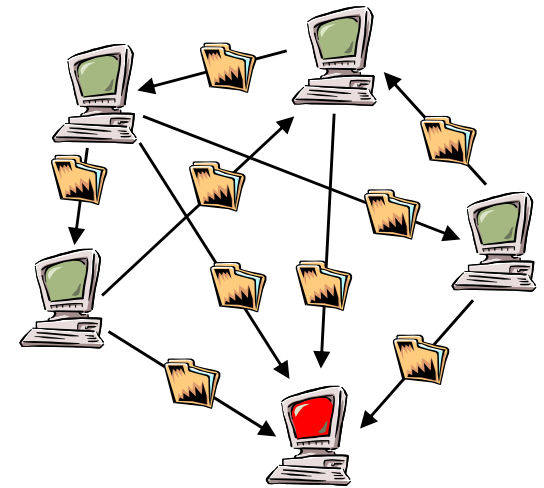
- Scalability, extensibility
- High performance
- Robustness (no Single Point of Failure)

□ Current weaknesses:

- Missing incentives (peers are autonomous!)
=> Vulnerability against selfish and malicious behavior
- Existing solutions are file sharing-specific
- Missing support for a commercial uses

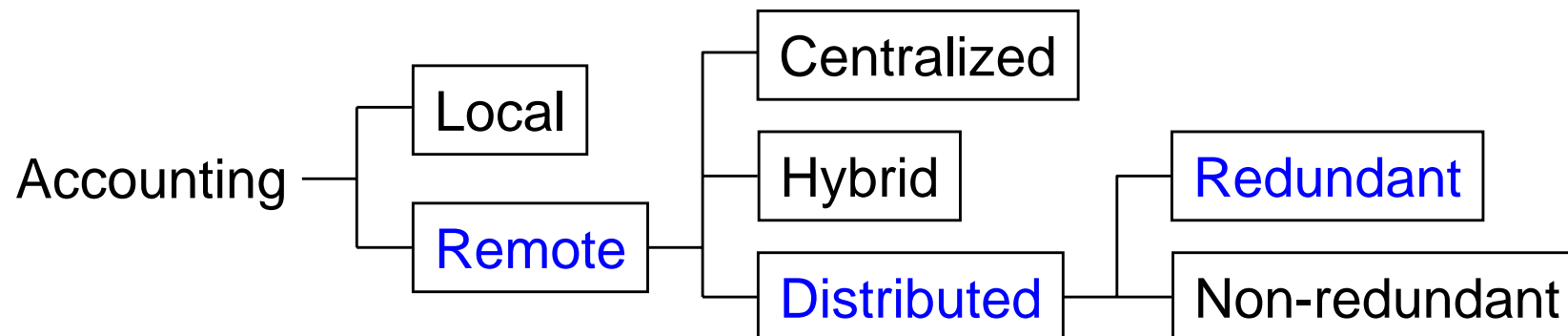
□ Objective:

- Design and implementation of a P2P accounting mechanism
 - Generic service support
 - Technically efficient and scalable
 - Reliable against malicious behavior
- => Goal: counter weaknesses, while keeping P2P benefits



Goals and Design Space of Accounting

- Ensure **accountability**
 - Account for the use of **services** or **resources** within an application
 - **Store** and **aggregate** accounting data over several transactions
- Provide **incentives**
 - **Keep track** of a peer's **balance** between contribution and consumption
=> Enforce fair sharing of resources by peers
- Enable **commercial use**
 - Serve as a basis for additional **charging** and **payment** mechanisms
- **Design space:**



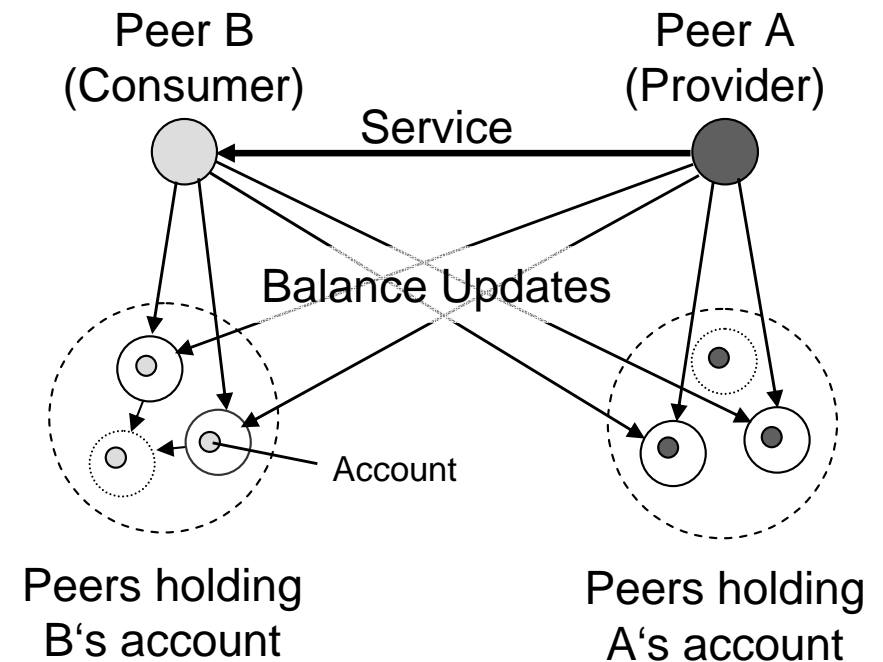
Distributed Redundant Accounting

□ Key Idea

- **Accounts** are **distributed** and **replicated** over all peers
 - Provider / consumer send **balance updates** to their account holders
- => Increases **reliability** and **availability** of the accounting data
- => A higher **credibility** or **trustworthiness** can be achieved

□ Problems

- **Collusion** among peers
 - Update only provider account
- **False-reporting** or **unreliable** peers
 - => Accounts become inconsistent
- Peers may **cheat**
 - Provider does not deliver service
 - Consumer sends incorrect balance update



PeerMint Approach

□ Sessions

- **Session** = use of a **service**, e.g. file download
- Represented by an **SLA** (signed by both session partners)

□ Accounts

- **Session accounts** keep accounting data within a session
 - Mapped onto m **session mediation peers**
 - Collect **balance updates** from provider and consumer
 - **Verify** if peers agree and update account accordingly
- **Peer accounts** aggregate a peers balance over several sessions
 - Mapped onto p **peer account holders**
 - Collect balance updates from **session mediation peers**
 - Account is updated according to **majority desicion**

□ Tariffs

- Specify when and by how much accounts are **updated**
- Included in **SLA**



PeerMint Technical Design

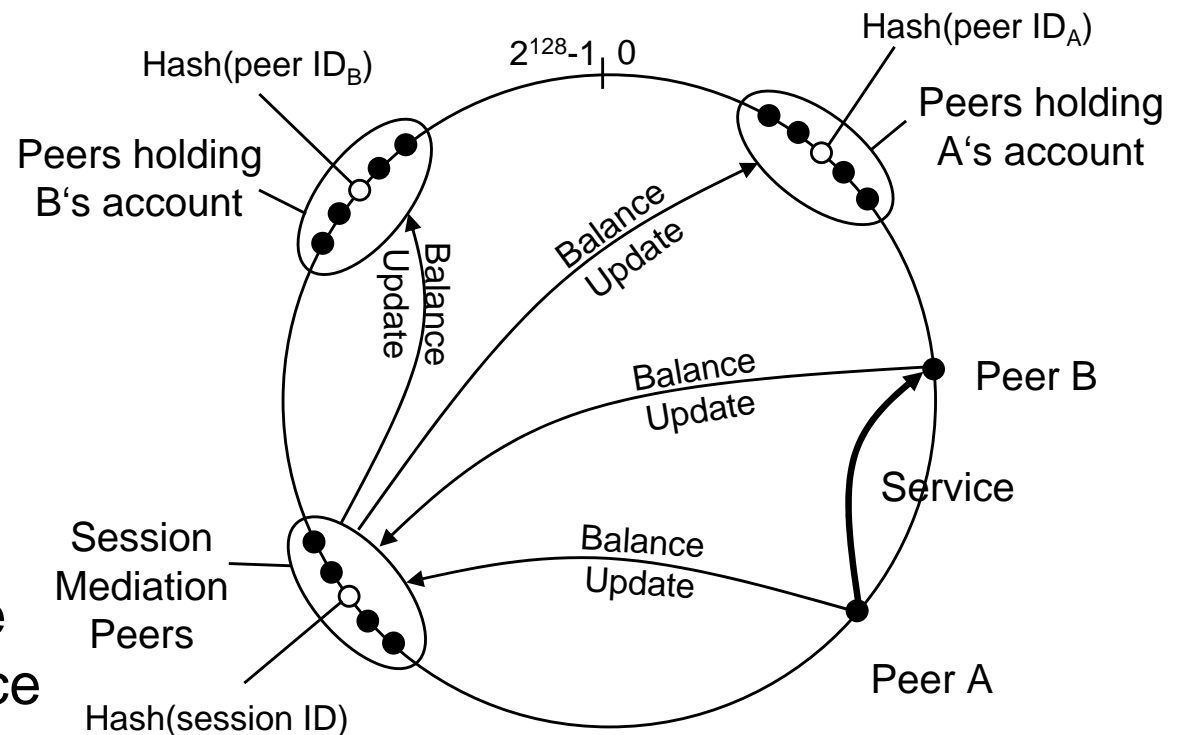
- Use of a structured P2P overlay network (Pastry)
 - Each peer has unique 128-bit `nodeId`
 - n closest `nodeIds` (leaf-set) stored in routing table
- Each peer has public / private key pair
 - Assumption: Public key certified offline and bound to `nodeId`

□ Mappings

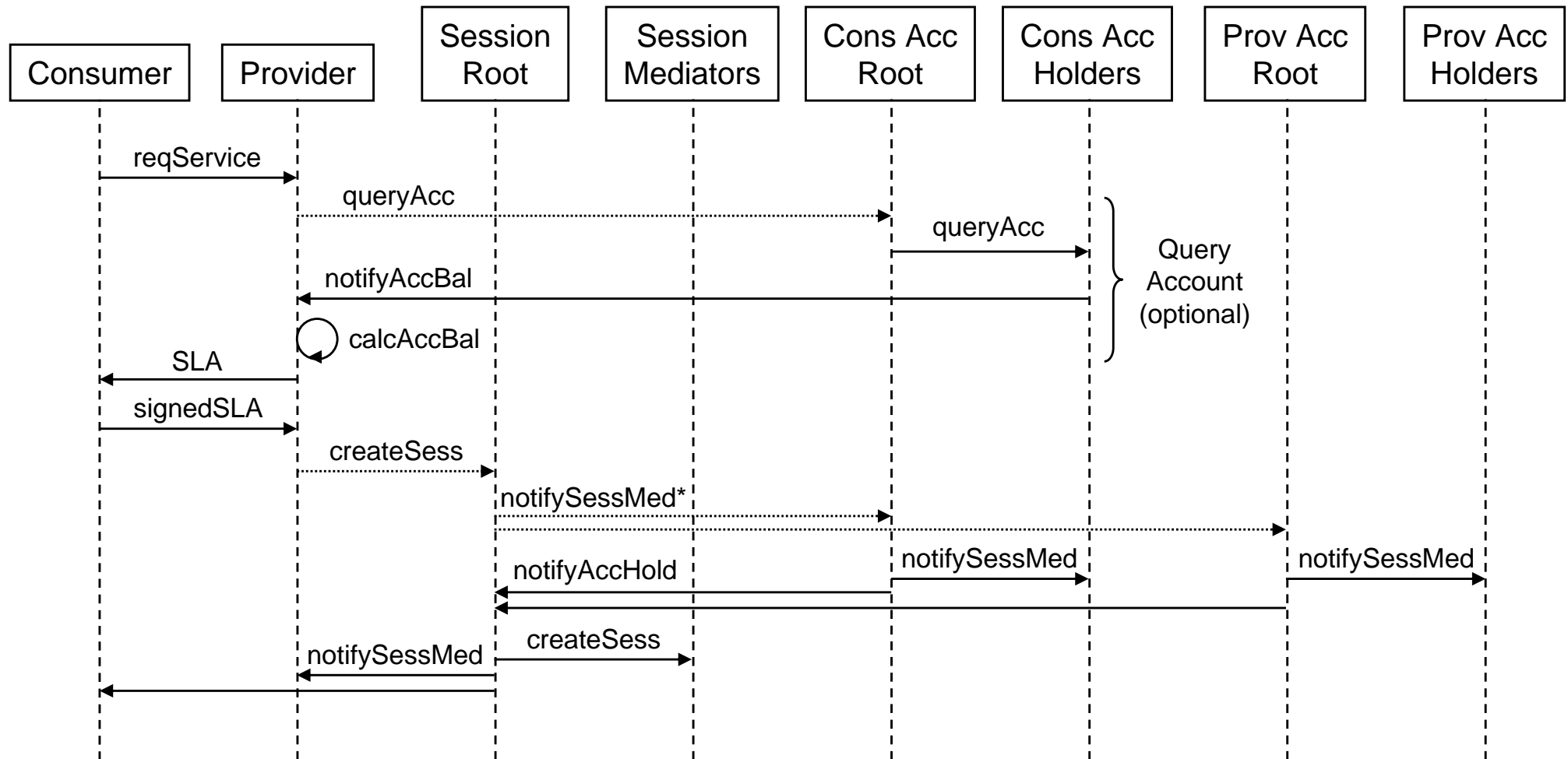
- Session account identified by `sessionId`
- Peer account identified by `peerId`

□ Account Maintenance

- **Leaf-set changes:** new peer becomes responsible and obtains current balance



Session Setup

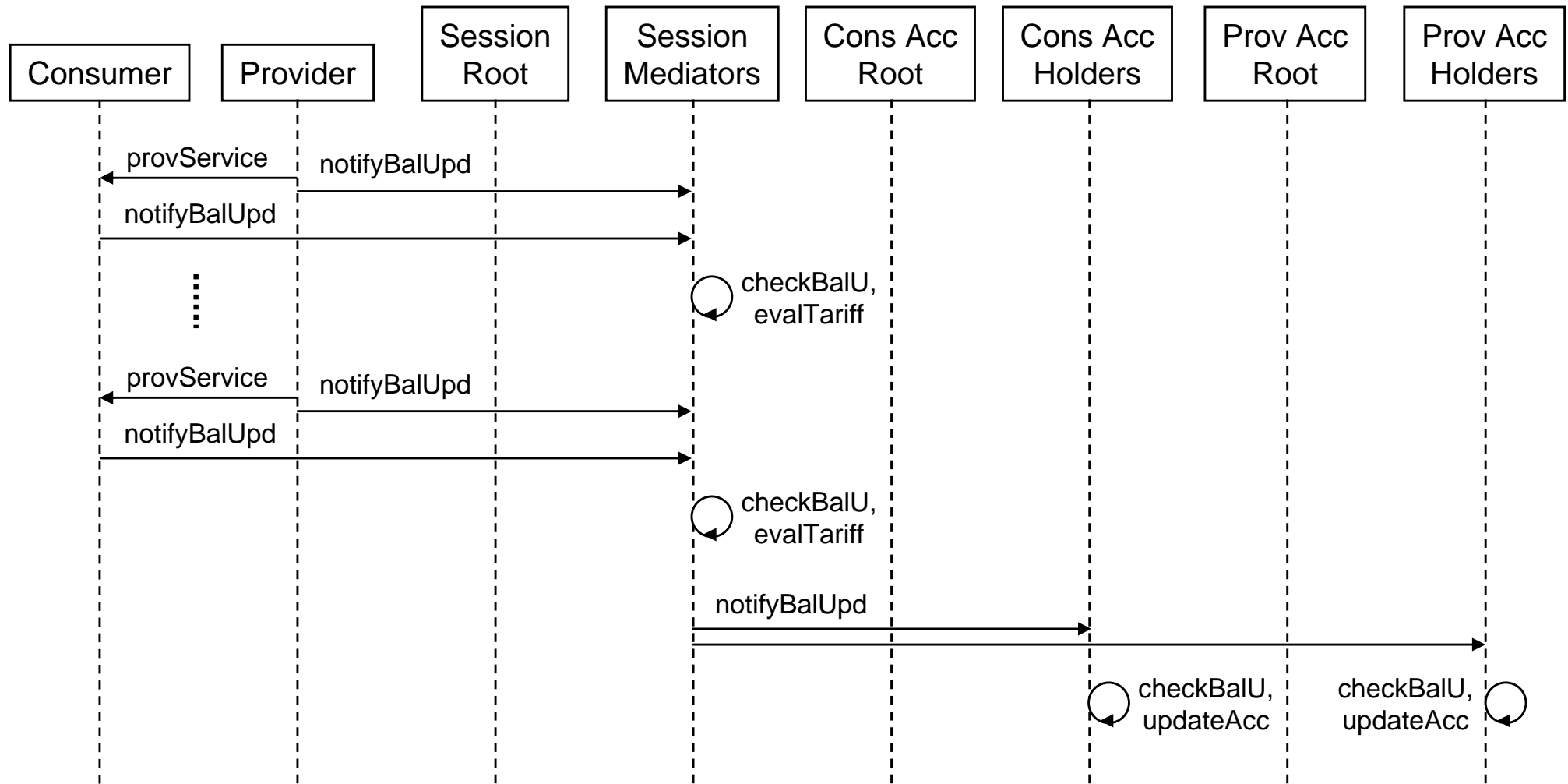


*) includes SLA to prove **eligibility**

.....> Pastry routing —> direct

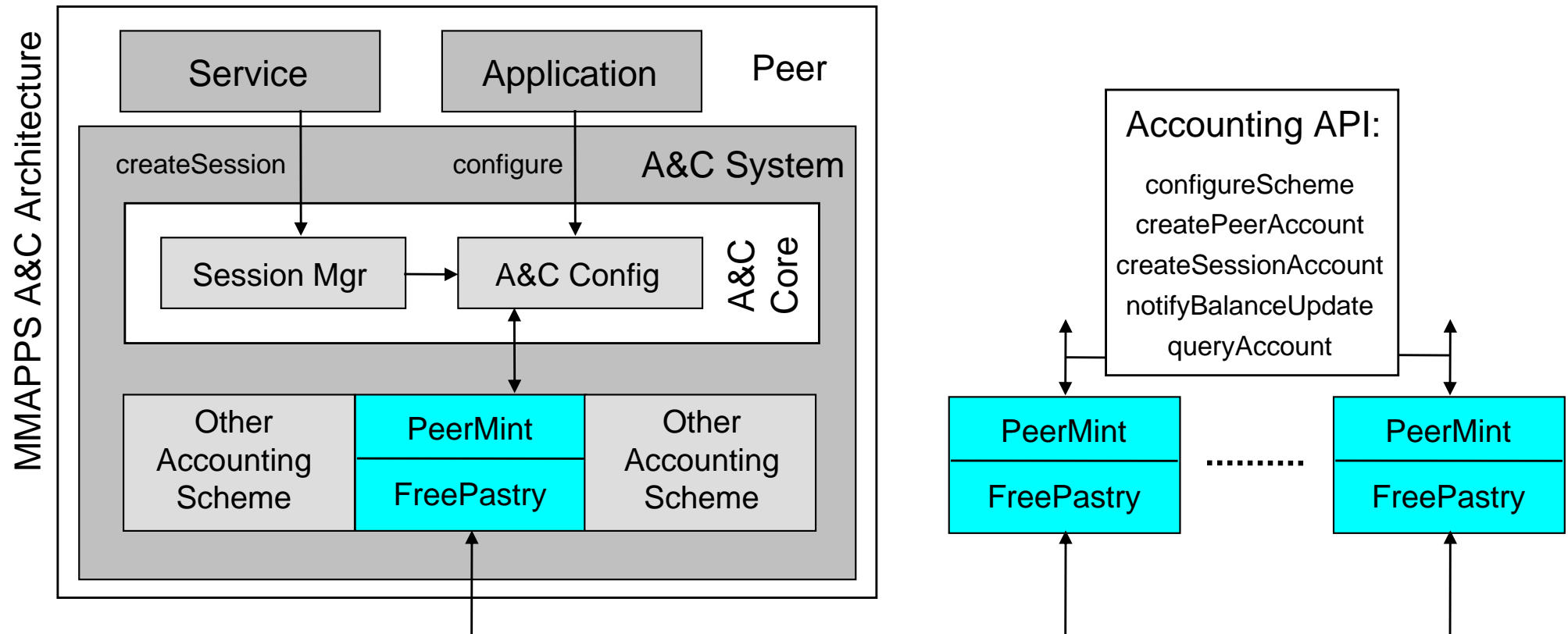


Account Balance Updates



Implementation

- PeerMint implemented into the **MMAPPS A&C system**
 - Embedded as an **accounting scheme** with a **generic accounting API**
- Prototype uses **FreePastry** as underlying P2P infrastructure
 - Open source implementation of Pastry in Java



Analytical Results: Efficiency

Costs	Cost driver	Peer Accounts	Session Accounts
Message costs	Account creation	$p + O(\log_b(N))$	$m + 1 + O(\log_b(N))$
	Account update	$2mp + O(\log_b(N))$	$2m$
	Account synchronization	$p - 1$	$m - 1$
	Account query	$2p + O(\log_b(N))$	$2m$
Storage costs	Avg. #accounts per peer	p / f	ms / f

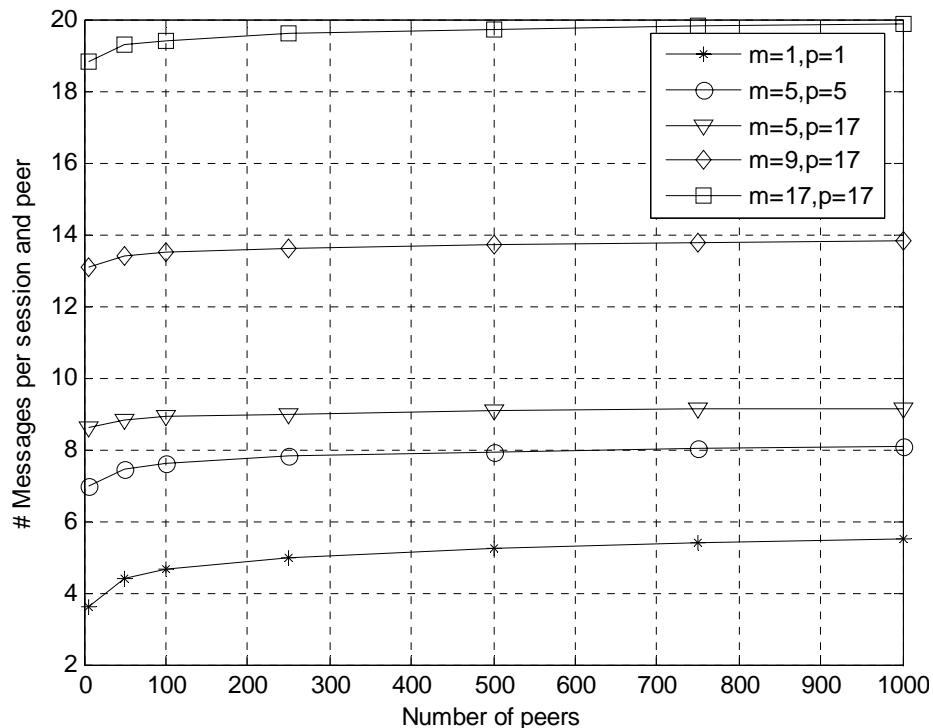
p: Number of peer account holders
 m: Number of session mediation peers
 N: Total number of peers
 s: Avg. number of ongoing sessions
 f: Avg. number of online peers

- Highest effort is needed to **update peer accounts**
 - However, peer account updates are less frequent than session account updates (depends on tariff)
- Note that there is a **basic overhead** for maintaining Pastry overlay network of $O(\log_b N)$
 - However, such an overlay network may already be present



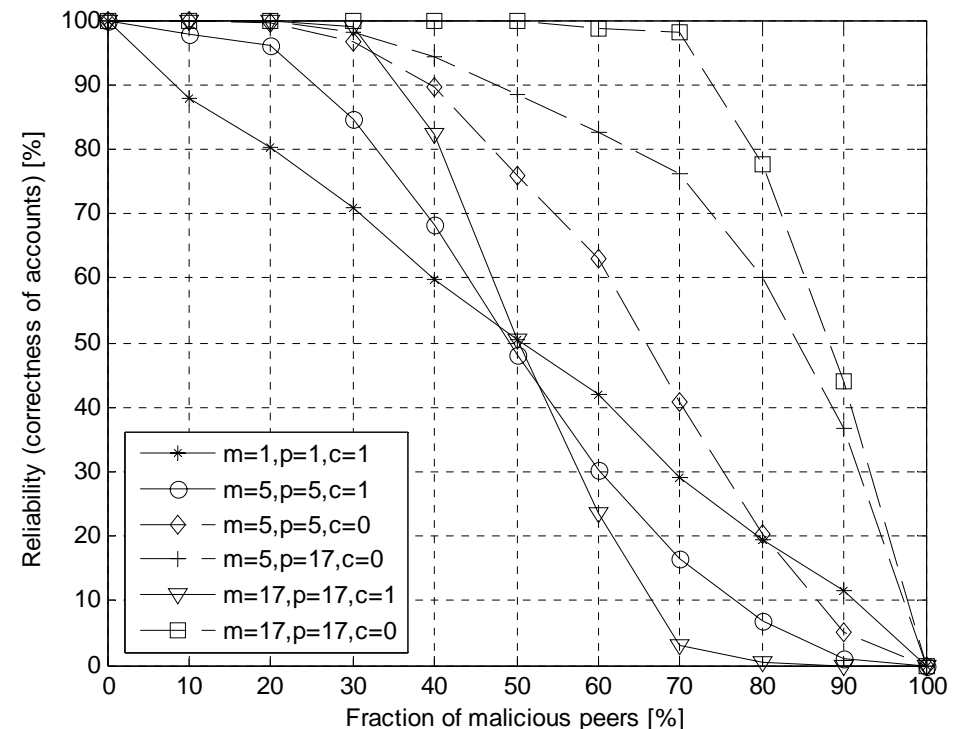
Experimental Results

Scalability, Efficiency



Malicious peers modeled as account holders reporting consistent ($c=1$) or arbitrary ($c=0$) false values (c relates to amount of collusion among malicious peers).

Reliability



- PeerMint correctly keeps accounts with $< 30\%$ malicious colluding peers by using 17 peer account holders and 17 session mediators.
- Without collusion, reliability can be achieved with $< 50\%$ malicious peers.



Conclusions and Future Work

- Designed mechanisms are **efficient** and **scale well**
- **High reliability** even in the presence of malicious peers
 - Achieved through redundancy
- **Generic** service support through use of **tariffs**
- Prototype implemented based on **FreePastry** and embedded into a generic **accounting and charging system**
- **Future Work**
 - Run prototype in a real-world environment (e.g. PlanetLab)
 - Consider and prevent overlay splitting
 - Use reputation to punish malicious peers
 - Study other forms of malicious attacks, e.g. DDoS



BACKUP

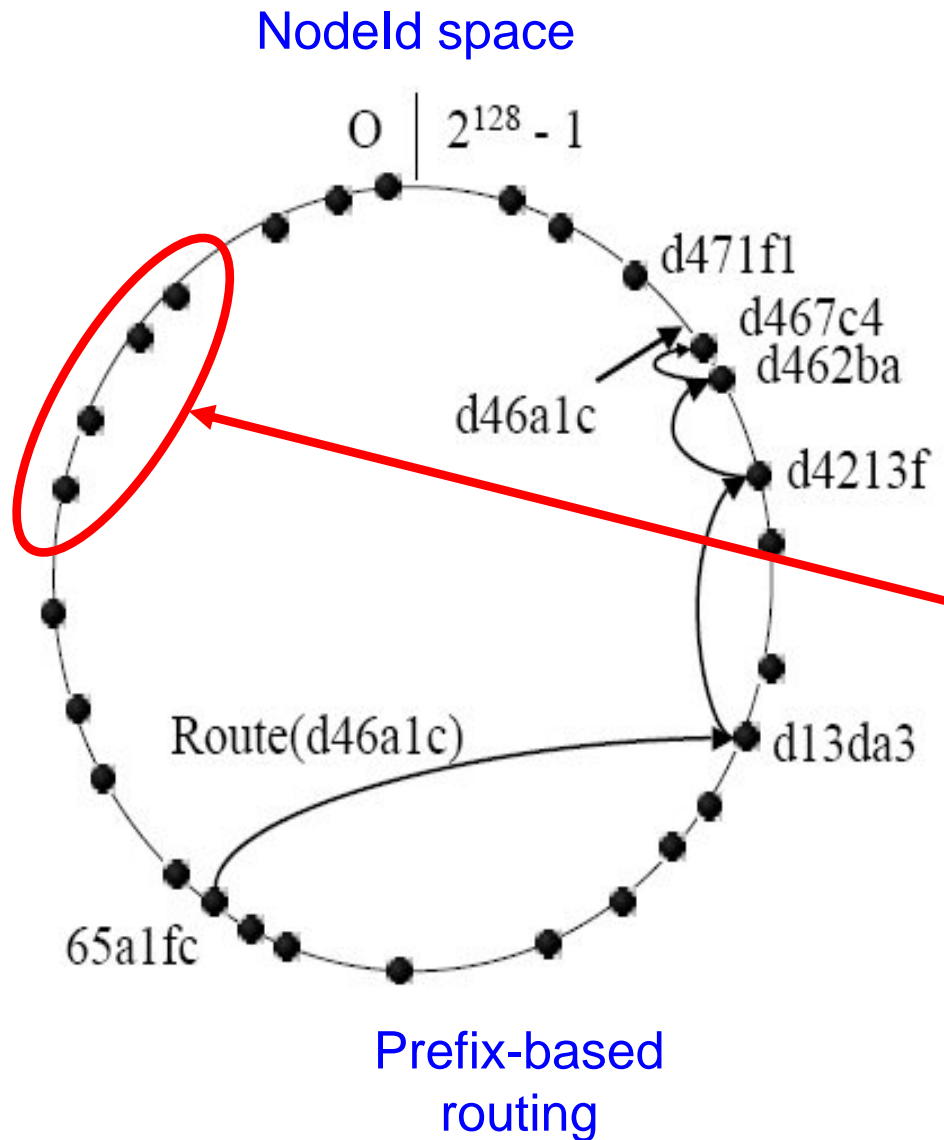


Related Work

- V. Vishnumurthy, S. Chandrakumar, E. Sirer: *KARMA: A Secure Economic Framework for Peer-to-Peer Resource*; P2P Economics, June 2003.
- B. Yang, H. Garcia-Molina: *PPay: Micropayments for Peer-to-Peer Systems*; CCS, October 2003.
- R. Dingledine, M. Freedman, D. Molnar: *Accountability*, Peer-To-Peer: Harnessing the Power of Disruptive Technologies, March 2001.
- B. Cohen: *Incentives Build Robustness in BitTorrent*; P2P Economics, June 2003.
- N. Liebau, V. Darlagiannis, A. Mauthe, R. Steinmetz: *Token-based Accounting for P2P-Systems*; KiVS 05, February 2005.



Underlying Infrastructure: Pastry



Unique 128-bit **nodelds** (calculated from a peers IP address or public key using secure hash function)

Open source implementation (**FreePastry**)

Routing Table

Nodeld 10233102

Leaf set	SMALLER	LARGER	
10233033	10233021	10233120	10233122
10233001	10233000	10233230	10233232

Routing table

-0-2212102	1	-2-2301203	-3-1203203
0	1-1-301233	1-2-230203	1-3-021022
10-0-31203	10-1-32102	2	10-3-23302
102-0-0230	102-1-1302	102-2-2302	3
1023-0-322	1023-1-000	1023-2-121	3
10233-0-01	1	10233-2-32	
0		102331-2-0	
		2	

Neighborhood set

13021022	10200230	11301233	31301233
02212102	22301203	31203203	33213321



Experiment Testbed

