TECHNISCHE UNIVERSITÄT DRESDEN

Trolli Schmittlauch

# Decentralised Hashtag Search and Subscription for Federated Social Networks

Prague, ActivityPubConf 2019

1970-01-01 Decentralised Hashtag Federation

introduce myself:
usually go by schmittlauch on the Internet
student of Computer Science @ TU Dresden
interest in federated systems and unusual social networks
presenting my work on a student research paper from this year

# Outline








Outline

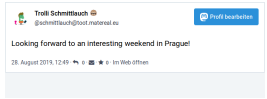
Motivation
Importance of #Hashtags
State of Hashtags in the Fediverse
System Architecture
Discussion
Social Considerations
Technical Considerations
Security Considerations
Summary

# Welcome to ActivityPubConf!
## Motivation






Welcome to ActivityPubConf!
Motivation




Who has been posting about this Conference?

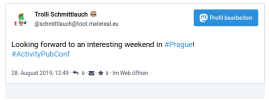
# Welcome to ActivityPubConf!
## Motivation







Welcome to ActivityPubConf!
Motivation


And who used #ActivityPubConf?

# Importance of #Hashtags

Hashtags are used for marking posts about certain topics or events:

- **events**: #ActivityPubConf, #CCCamp19






Importance of #Hashtags
Hashtags are used for marking posts about certain topics or events:
- **events**: #ActivityPubConf, #CCCamp19
ActivityPub Conference

mark topics of posts, make them discoverable by content. No decentralised full text search in Fediverse (centralised search engines)

# Importance of #Hashtags

Hashtags are used for marking posts about certain topics or events:

- **events**: #ActivityPubConf, #CCCamp19
- **political topics**: #SaveTheInternet


"Obama in the Backseat: Rally to Save the Internet" by Free Press Pics is licensed under CC BY-SA 2.0





Importance of #Hashtags
Hashtags are used for marking posts about certain topics or events:
• events: #ActivityPubConf, #CCCamp19
• political topics: #SaveTheInternet

mark topics of posts, make them discoverable by content. No decentralised full text search in Fediverse (centralised search engines)

# Importance of #Hashtags

Hashtags are used for marking posts about certain topics or events:

- **events**: #ActivityPubConf, #CCCamp19
- **political topics**: #SaveTheInternet
- **general topics**: #mastoadmin, #Tusky





Importance of #Hashtags
Hashtags are used for marking posts about certain topics or events:
- **events**: #ActivityPubConf, #CCCamp19
- **political topics**: #SaveTheInternet
- **general topics**: #mastoadmin, #Tusky

mark topics of posts, make them discoverable by content. No decentralised full text search in Fediverse (centralised search engines)

# Importance of #Hashtags

Hashtags are used for marking posts about certain topics or events:

- **events**: #ActivityPubConf, #CCCamp19
- **political topics**: #SaveTheInternet
- **general topics**: #mastoadmin, #Tusky
- **ongoing demonstrations**: #GeziPark, #WomensMarch







Importance of #Hashtags
Hashtags are used for marking posts about certain topics or events:
events: #ActivityPubConf, #CCCamp19
political topics: #SaveTheInternet
general topics: #mastoadmin, #Tusky
ongoing demonstrations: #GeziPark, #WomensMarch



mark topics of posts, make them discoverable by content. No decentralised full text search in Fediverse (centralised search engines)

# Importance of #Hashtags

Hashtags are used for marking posts about certain topics or events:

- **events**: #ActivityPubConf, #CCCamp19
- **political topics**: #SaveTheInternet
- **general topics**: #mastoadmin, #Tusky
- **ongoing demonstrations**: #GeziPark, #WomensMarch
- **social movements**: #MeToo








Importance of #Hashtags
Hashtags are used for marking posts about certain topics or events:
events: #ActivityPubConf, #CCCamp19
political topics: #SaveTheInternet
general topics: #mastoadmin, #Tusky
ongoing demonstrations: #GeziPark, #WomensMarch
social movements: #MeToo

mark topics of posts, make them discoverable by content. No decentralised full text search in Fediverse (centralised search engines)

# State of Hashtags on the Fediverse

Hashtags are used in the Fediverse

# State of Hashtags on the Fediverse

Hashtags are used in the Fediverse

But do they behave as expected?





**State of Hashtags on the Fediverse**

Hashtags are used in the Fediverse

But do they behave as expected?


Figure: #activitypubconf on the single-user instance *toot.matereal.eu*


Figure: #activitypubconf on the large instance *mastodon.social*






Figure: #activitypubconf on the single-user instance *toot.matereal.eu*


Figure: #activitypubconf on the large instance *mastodon.social*

# State of Hashtags on the Fediverse
## Fragmentation

- fragmented view on hashtag posts depending on user's instance
- hashtag search only on locally known posts
- Result: incentive to cluster on large nodes ⇐ centralisation



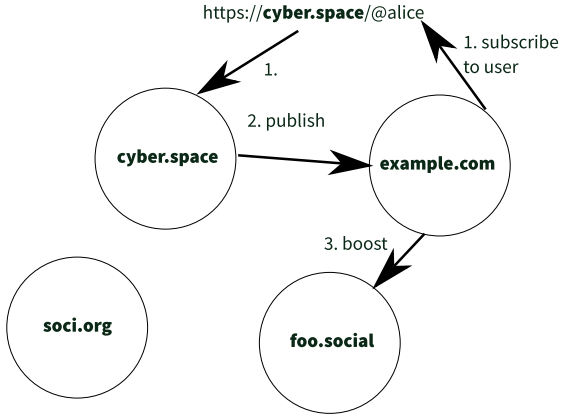


**State of Hashtags on the Fediverse**
Fragmentation

- fragmented view on hashtag posts depending on user's instance
- hashtag search only on locally known posts
- Result: incentive to cluster on large nodes ⇐ centralisation

1. view depends on users instance
2. local posts
3. cluster incentive

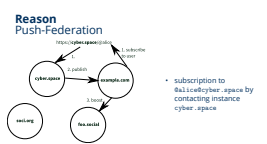
# Reason
## Push-Federation




- subscription to `@alice@cyber.space` by contacting instance `cyber.space`

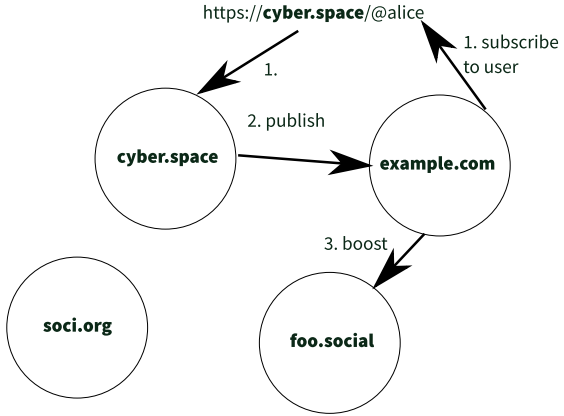







example scenario with 4 instances

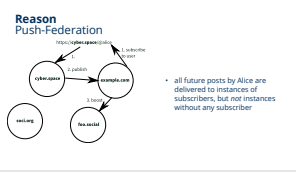
# Reason
## Push-Federation




- all future posts by Alice are delivered to instances of subscribers, but *not* instances without any subscriber



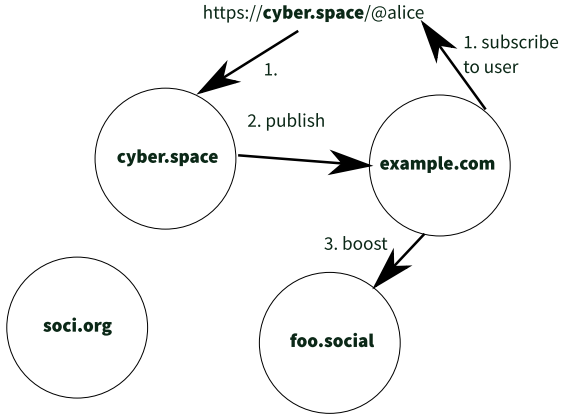





example scenario with 4 instances

1. cyber.space may not even be aware of existence of other instances

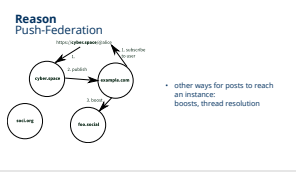
# Reason
## Push-Federation




- other ways for posts to reach an instance:
  boosts, thread resolution








example scenario with 4 instances

1. posts can disseminate through other means

# Current Solutions

- Mastodon PubRelay or Pleroma lite-pub relay:
  - centralised actor relaying all incoming posts
  - single point of failure, which relay to choose?
  - relaying all incoming posts ⇒ huge load on small instances
  - only access to posts sent after initial subscription
- Diaspora* SocialRelay
  - similar, but allows subscribing to certain tags only





Current Solutions

- Mastodon PubRelay or Pleroma lite-pub relay:
  - centralised actor relaying all incoming posts
  - single point of failure, which relay to choose?
  - relaying all incoming posts ⇒ huge load on small instances
  - only access to posts sent after initial subscription
- Diaspora* SocialRelay
  - similar, but allows subscribing to certain tags only

relays
centralised actors, single point
which one to choose?
all posts -> overload
after subscription

# System Architecture
## Goals

- **relay & subscribe**: instances can subscribe to all public posts of a hashtag
- **store & query**: instances can retrieve 1 year of history for a hashtag without subscription
- fully decentralised, no single point of authority for all tags





2 goals: relay & subscribe; store & query; fully decentralised

System Architecture
Goals
- relay & subscribe: instances can subscribe to all public posts of a hashtag
- store & query: instances can retrieve 1 year of history for a hashtag without subscription
- fully decentralised, no single point of authority for all tags

# System Architecture
## adding a DHT backend to the Fediverse

core idea: distribute responsibility for tags among instances using a **D**istributed **H**ash **T**able,
based on Chord [1]





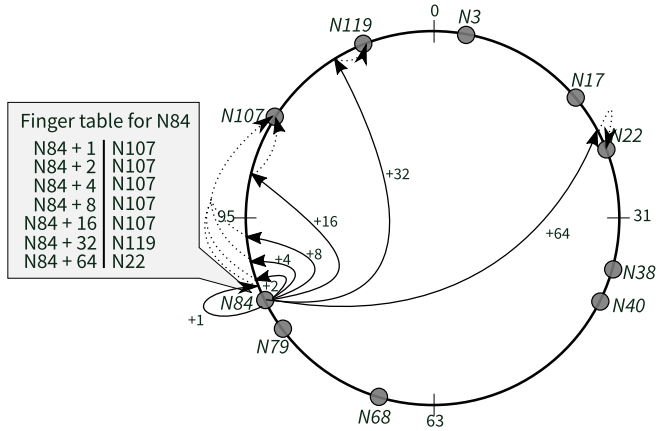
**System Architecture**
adding a DHT backend to the Fediverse

core idea: distribute responsibility for tags among instances using a **D**istributed **H**ash **T**able,
based on Chord [1]

1. subscription to all posts of a user possible because there is a single responsible instance
2. distribute responsibility for posts of a hashtag = relaying & storage
3. DHT: structured P2P networks, self-organising, no central authority
4. provides efficient (log N) key-value storage and lookup

# System Architecture
## adding a DHT backend to the fediverse

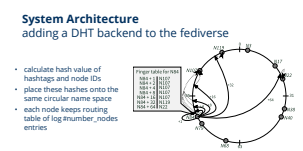
- calculate hash value of hashtags and node IDs
- place these hashes onto the same circular name space
- each node keeps routing table of log #number_nodes entries





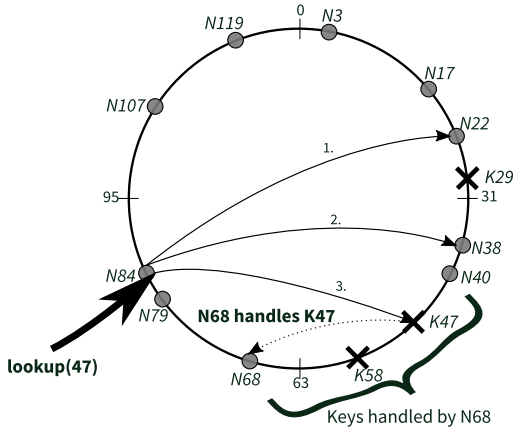



1. common namespace for nodes and lookup keys
2. routing table entries to distances of powers of 2



# System Architecture
## adding a DHT backend to the fediverse

- responsible node for a key k is node successor(k) = $\min_i(k + i) \mod keyspace\_size$
- DHT used for iterative lookup of responsible relay/ storage node
- my architecture: keyspace = $2^{256}$ with 256bit-long IDs

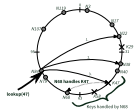







1. iterative lookup of responsible successor node of key
2. i use keyspace of $2^{256}$


System Architecture
adding a DHT backend to the fediverse



# Publishing, Relaying and Storage
## lifecycle of posts

1. publishing instance looks up responsible relay instance on DHT for each included hashtag
2. publishing instance sends post to responsible relay instance
3. relay instance looks up responsible storage node on DHT
4. relay instance verifies incoming post's signature, then relays post URI (ID) to all subscribers + storage node
5. subscribing instances can now retrieve the full authenticated post from received post URI





**Publishing, Relaying and Storage**
lifecycle of posts

1. publishing instance looks up responsible relay instance on DHT for each included hashtag
2. publishing instance sends post to responsible relay instance
3. relay instance looks up responsible storage node on DHT
4. relay instance verifies incoming post's signature, then relays post URI (ID) to all subscribers + storage node
5. subscribing instances can now retrieve the full authenticated post from received post URI

1. calculate hashum of hashtag -> lookup key for DHT
2. only post ID relayed, but not full post content. Reasons: LDSignatures not supported everywhere, deniability & revocation
3. for joining and leaving the DHT see paper

- node ID determines set of hashtags handled by instance
- problem: for security reasons, node **must not** choose their IDs freely
- Can instances be overloaded by their assigned hashtag posts?

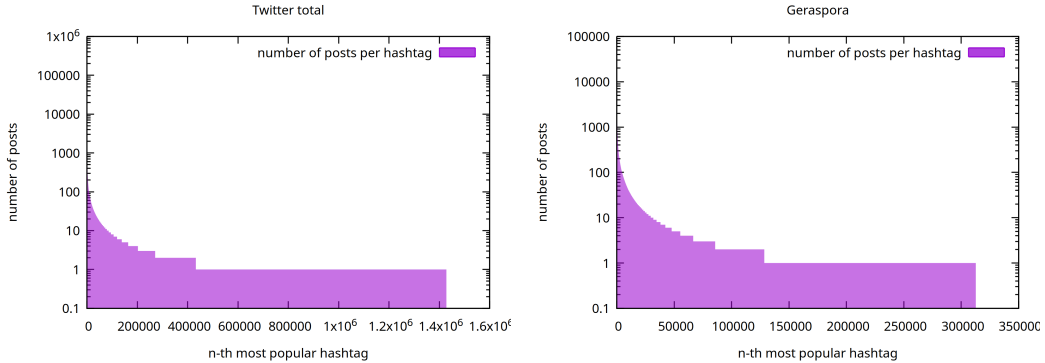




- node ID determines set of hashtags handled by instance
- problem: for security reasons, node **must not** choose their IDs freely
- Can instances be overloaded by their assigned hashtag posts?

so far so easy.
what hashtags does a certain instance handle? determined by node ID
can it bear the load?

# Distribution of Posts per Tag




distribution of posts per hashtag follows a steep power law






Distribution of Posts per Tag
distribution of posts per hashtag follows a steep power law


analysis of a 1 month dump of Twitter, Geraspora (Diaspora) and Friendica posts
Twitter: 70% of hashtags used just once
note the logarithmic axis!So what if a small node gets several large hashtags? => need for load balancing

# Load Balancing
## of hashtags between nodes

- *k-choices* algorithm by Ledlie and Seltzer [2]
- a node can represent several *virtual nodes* on the DHT
- $\kappa$ possible virtual node IDs: $ID = \text{hash}(ID' + +i), i \in \{0, 1, \ldots, \kappa - 1\}$
- nodes have a **capacity** and choose set of active IDs according to lowest mismatch of own and neighbour node capacity
- querying load of potential IDs before joining, periodic re-balancing
- independent load balancing of relay and storage nodes due to independent DHTs





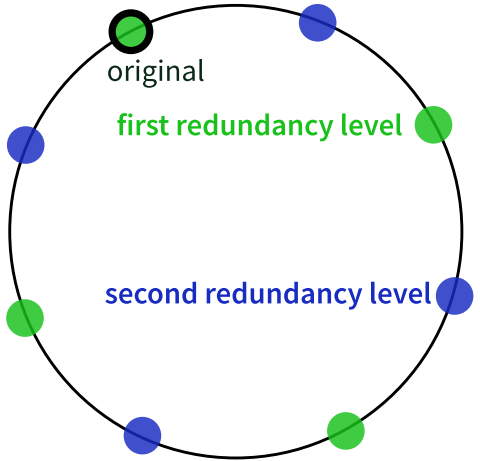
**Load Balancing**
of hashtags between nodes

- *k-choices* algorithm by Ledlie and Seltzer [2]
- a node can represent several *virtual nodes* on the DHT
- $\kappa$ possible virtual node IDs: $ID = \text{hash}(ID' + +i), i \in \{0, 1, \ldots, \kappa - 1\}$
- nodes have a **capacity** and choose set of active IDs according to lowest mismatch of own and neighbour node capacity
- querying load of potential IDs before joining, periodic re-balancing
- independent load balancing of relay and storage nodes due to independent DHTs

1. enable different roles “relay”, “storage” to balance independently: 2 DHTs
2. a simple simulation on the effectivity of the balancing algorithm can be found in the paper

# Redundancy

- redundant assignment of responsibility for hashtag at equal distances on Chord ring, inspired by Harvesf and Blough [3]
- default redundancy: $2^2 = 4$, scalable in powers of 2
- **relay nodes**: hot standby nodes take over in overload situations (load spikes)
- **storage nodes**: overloaded nodes can split stored posts by content hash and double redundancy set








resilience against node failure, allows data validation through cross-checking

# Discussion
## I need YOUR feedback

I want feedback from all of you, no matter whether it's from a *technical* or from a *social perspective*.





**Discussion**
I need YOUR feedback

I want feedback from all of you, no matter whether it's from a *technical* or from a *social perspective*.

architecture just a concept so far
before implementations: several open questions

# Social Considerations

Do we even want global hashtags in the Fediverse?

- positive potential (conversation, coordination) vs. negative potential (spam, harrassment)
- visibility level: public posts only, unlisted, new level necessary?
- relaying post URI only should provide plausible deniability and retractability





1. positive vs negative potential
2. visibility levels
3. retractability should be given

Social Considerations

Do we even want global hashtags in the Fediverse?
- positive potential (conversation, coordination) vs. negative potential (spam, harrassment)
- visibility level: public posts only, unlisted, new level necessary?
- relaying post URI only should provide plausible deniability and retractability

# Technical Considerations
## instance admins

- intended as opt-in, domain-based push federation still better for user subscriptions
- assumption: instances offer 5.5× the storage & 2.5× the bandwidth of own posts
- performance: Can this be implemented efficiently enough to not DDoS popular hashtag nodes?
  - batched retrieval of posts from same source
  - exponential backoff retries





**Technical Considerations**
instance admins

- intended as opt-in, domain-based push federation still better for user subscriptions
- assumption: instances offer 5.5x the storage & 2.5x the bandwidth of own posts
- performance: Can this be implemented efficiently enough to not DDoS popular hashtag nodes?
  - batched retrieval of posts from same source
  - exponential backoff retries

1. optional mechanism, let's keep C2S communication for mobile friendliness and PushFed for simplicity
2. assumptions: 5.5x storage, 2.5x bandwidth
3. performance: fetch DDoS of popular tags

# Technical Considerations
## integration into the ActivityPub Fediverse

- This architecture is an unimplemented concept so far!
- integration into ActivityPub ecosystem
  - hashtags may be represented as relay actors with own in- & outbox, addressed in cc
  - relaying to subscribers via SharedInbox
  - idea for addressing: new URI scheme that gets transparently resolved to responsible node's domain via DHT by application proxy
  - signalling of error codes and redundancy factors is needed
- DHT routing communication does not use ActivityPub





**Technical Considerations**
integration into the ActivityPub Fediverse

- This architecture is an unimplemented concept so far!
- integration into ActivityPub ecosystem
  - hashtags may be represented as relay actors with own in- & outbox, addressed in cc
  - relaying to subscribers via SharedInbox
  - idea for addressing: new URI scheme that gets transparently resolved to responsible node's domain via DHT by application proxy
  - signalling of error codes and redundancy factors is needed
- DHT routing communication does not use ActivityPub

1. disclaimer: I'm new to ActivityPub and have no implementation experience
2. application proxy for transparent URI scheme resolving?

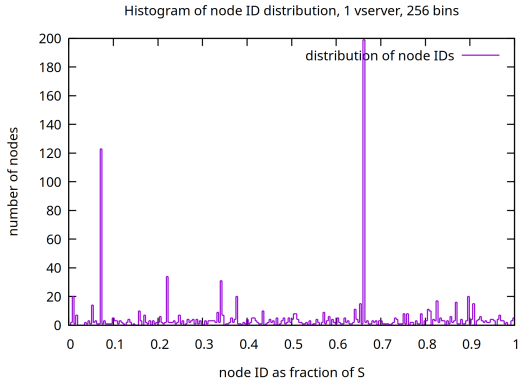
# Technical Considerations
## node ID assignment





1. Let's talk about the elephant in the room of "federated services"
2. common DHT attack: Sybil-… = 1 attacker introduces large number of nodes
3. sorry to all instance admins, but: CloudFlare behaves like a MITM/ Sybil attacker
4. node ID derivation: first 64 bits of IPv6

# Technical Considerations
## node ID assignment

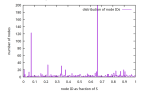



1st peak: Masto.host, 2nd peak: Cloudflare






Technical Considerations
node ID assignment


1st peak: Masto.host, 2nd peak: Cloudflare


1. Let's talk about the elephant in the room of "federated services"
2. common DHT attack: Sybil-… = 1 attacker introduces large number of nodes
3. sorry to all instance admins, but: CloudFlare behaves like a MITM/ Sybil attacker
4. node ID derivation: first 64 bits of IPv6

# Technical Considerations
## node ID assignment

$$h_n = \texttt{hash(IPv6\_addr[0,63] ++ vserver)[0,63]} \\ \texttt{++ hash(domain ++ vserver)[0,127]} \\ \texttt{++ hash(IPv6\_addr[0,63] ++ vserver)[64,127]}$$

node ID derivation

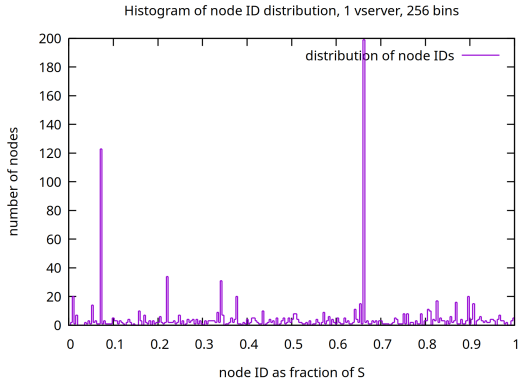




**Technical Considerations**
node ID assignment
$h_n$ = hash(IPv6_addr[0,63] ++ vserver)[0,63] ++ hash(domain ++ vserver)[0,127] ++ hash(IPv6_addr[0,63] ++ vserver)[64,127]
node ID derivation

1. Let's talk about the elephant in the room of "federated services"
2. common DHT attack: Sybil-... = 1 attacker introduces large number of nodes
3. sorry to all instance admins, but: CloudFlare behaves like a MITM/ Sybil attacker
4. node ID derivation: first 64 bits of IPv6

# Technical Considerations
## node ID assignment

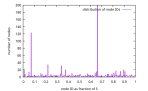



1st peak: Masto.host, 2nd peak: Cloudflare






Technical Considerations
node ID assignment


1st peak: Masto.host, 2nd peak: Cloudflare


1. Let's talk about the elephant in the room of "federated services"
2. common DHT attack: Sybil-… = 1 attacker introduces large number of nodes
3. sorry to all instance admins, but: CloudFlare behaves like a MITM/ Sybil attacker
4. node ID derivation: first 64 bits of IPv6

# Security Considerations

- attacker shall not be able to deliberately gain responsibility for certain hashtags
  - node ID mainly dependant on IPv6 subnet
- attacker shall not introduce arbitrary number of nodes
  - valid domain required for node ID derivation, assumption: domains cost money





Security Considerations

- attacker shall not be able to deliberately gain responsibility for certain hashtags
  - node ID mainly dependant on IPv6 subnet
- attacker shall not introduce arbitrary number of nodes
  - valid domain required for node ID derivation, assumption: domains cost money

not perfectly secure, but the best I could think of. Better ideas welcome

# Summary

- decentralised architecture for handling posts of the same hashtag:
  - subscribe to hashtag and get posts relayed
  - query stored posts of a certain hashtag without subscription
- responsibility for hashtag divided among instances using a DHT
- architecture balances the load between nodes and maintains redundancy
- several open questions before implementation





Summary

- decentralised architecture for handling posts of the same hashtag:
  - subscribe to hashtag and get posts relayed
  - query stored posts of a certain hashtag without subscription
- responsibility for hashtag divided among instances using a DHT
- architecture balances the load between nodes and maintains redundancy
- several open questions before implementation

# Questions, comments, feedback?

https://git.orlives.de/schmittlauch/paper_hashtag_federation/src/branch/master/paper_hashtag_federation.pdf

@schmittlauch@toot.matereal.eu





Questions, comments, feedback?
https://git.orlives.de/schmittlauch/paper_hashtag_federation/src/branch/master/paper_hashtag_federation.pdf
@schmittlauch@toot.matereal.eu

# References I


- I. Stoica et al. “Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications”. In: *IEEE/ACM Transactions on Networking* 11.1 (Feb. 2003), pp. 17–32. ISSN: 1063-6692. DOI: 10.1109/TNET.2002.808407.
- J. Ledlie and M. Seltzer. *Harvard Technical Report TR-31-04: Distributed, Secure Load Balancing with Skew, Heterogeneity, and Churn*. Dec. 2004. URL: https://people.csail.mit.edu/ledlie/lb/kchoices05-tr.pdf (visited on 10/07/2019).
- C. Harvesf and D. M. Blough. “The Effect of Replica Placement on Routing Robustness in Distributed Hash Tables”. In: *Sixth IEEE International Conference on Peer-to-Peer Computing (P2P’06)*. Sixth IEEE International Conference on Peer-to-Peer Computing (P2P’06). Sept. 2006, pp. 57–6. DOI: 10.1109/P2P.2006.44.







References I

I. Stoica et al. “Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications”. In: IEEE/ACM Transactions on Networking 11.1 (Feb. 2003), pp. 17–32. ISSN: 1063-6692. DOI: 10.1109/TNET.2002.808407.

J. Ledlie and M. Seltzer. Harvard Technical Report TR-31-04: Distributed, Secure Load Balancing with Skew, Heterogeneity, and Churn. Dec. 2004. URL: https://people.csail.mit.edu/ledlie/lb/kchoices05-tr.pdf (visited on 10/07/2019).

C. Harvesf and D. M. Blough. “The Effect of Replica Placement on Routing Robustness in Distributed Hash Tables”. In: Sixth IEEE International Conference on Peer-to-Peer Computing (P2P’06). Sixth IEEE International Conference on Peer-to-Peer Computing (P2P’06). Sept. 2006, pp. 57–6. DOI: 10.1109/P2P.2006.44.