Resilient, scalable services on top of P2P storage networks

Marios Isaakidis

eQualitie - University College London

marios@equalit.ie

7DD5 D164 B690 0A9D FEE2 98B5 E977 D7B8 5232 4D27

December 29, 2015

Overview

What can be built?

Anonymous Communication Channels
- Killer Features
- Existing Services

Anonymous Communication Channels
- Needed Key Abstractions
- Establishing a channel
- Speed Hacks and Scaling

Backbone Nodes

Last thoughts

What can be built?

Killer Features

- Once inserted, your content will be available even if your node is not online
- No overwrites/removals, but self-revisioned
- Easy to bootstrap
- Anonymity
- The more the users the more resilient the service becomes
- Truly non-hierarchical $\Rightarrow$ no central point of failure, no need for knowing special nodes in the network

But: static content only - all operations are either insertions or retrievals

Existing Services

Cool applications:

- Pseudo-Identities (Web of Trust)
  - Really useful for discovering other users and blocking spammers
- Messaging (Freemail, FLIP-IRC)
- Twitter (Sone)
- Forums (Frost, Freetalk)
- DNS

Didn't catch on, because we have more suitable systems (onion/garlic routing, mix networks...)

Existing Services

Self-archiving:

- Censorship-circumvention (CENO - https://censorship.no)
  - No need for a priori knowledge of an "exit node"
  - In case of nationwide throttling, content remains available
  - If a website is taken down, its cached version exists in the network
  - Request for caching a website are handled by the service only once; then served via the distributed cache
- Wikis
- Version control (Infocalypse)
- TODO Whistleblowing/leaking platforms

Insert interesting content!

Anonymous Communication Channels

Needed Key Abstractions

- CHK - Content Hash keys
  - Files are split into chunks, content is hashed and this is what you get
  - Base keys, used for inserting and requesting file parts
- SSK - Signed Space keys
  - Public-key cryptography based keys (we will be referring to them as Private and Public SSK)
  - Only the owner of them can insert content (as well as newer versions) with those keys
- KSK - Keyword-Signed keys
  - Deterministically get mapped to a private SSK

Establishing a channel

1. Service provider announces its Public RSA key and a Quiz
2. Clients who know the public SSK of the Service Provider retrieve the announcement
3. Solving the puzzle results to a number of KSK slots
4. Clients generate a private SSK key and encrypt it using the Service Provider's RSA key, then insert it into one of the KSK slots
5. Service provider will eventually poll the KSK slots and decrypt the content, threfore learn about the private SSK inserted by a client
6. Service provider inserts a "SYN" message using that private SSK and client is using the same channel to make requests
7. Once the slots are consumed, Service Provider makes a new announcement with a new Quiz

Speed Hacks and Scaling

Establishing a channel requires 2 retrievals and 2 insertions
Sending a message requires only 1 insertion and 1 retrieval

- Speed hacks:
  - insert the content in the metadata
  - start sending messages to the service provider before retrieving the "SYN" response
- CAPTCHAs can be used as the Quizes to stop bots from spamming the KSK slots
- Distribute KSK-polling to various nodes
- Distribute established channels to various nodes to handle requests by users
- If a channel is inactive for some time, stop polling it for new messages
- DDoS resistant by design - spammers can only spam their own channels, which then we can stop polling for updates

Backbone Nodes

aka Small World Routing

Backbone Nodes

- Nodes that do not have a particular task, apart from connecting with high trust links with our special nodes
- allocated high bandwidth and storage limits
- If they go down it's still ok
- Darknet - they hide our nodes from the rest of the network
- Protect our nodes from de-anonymization attacks
- Distributed in various areas of the DHT, they forward requests for retrievals/insertions much faster
- They are caching the Service Provider's content therefore increasing its longevity-availability

Last thoughts

- Scaling is achieved by distributing tasks to different types of nodes
- System designers should try to build modular functionality
- Communication between the nodes we control can be done using a direct N2N mechanism
- Pre-emptively insert content, present what is already in the distributed cache
- Keep the P2P network healthy