RobustIRC
oder: IRC ohne Netsplits

Easterhegg 2016, 2016-03-27

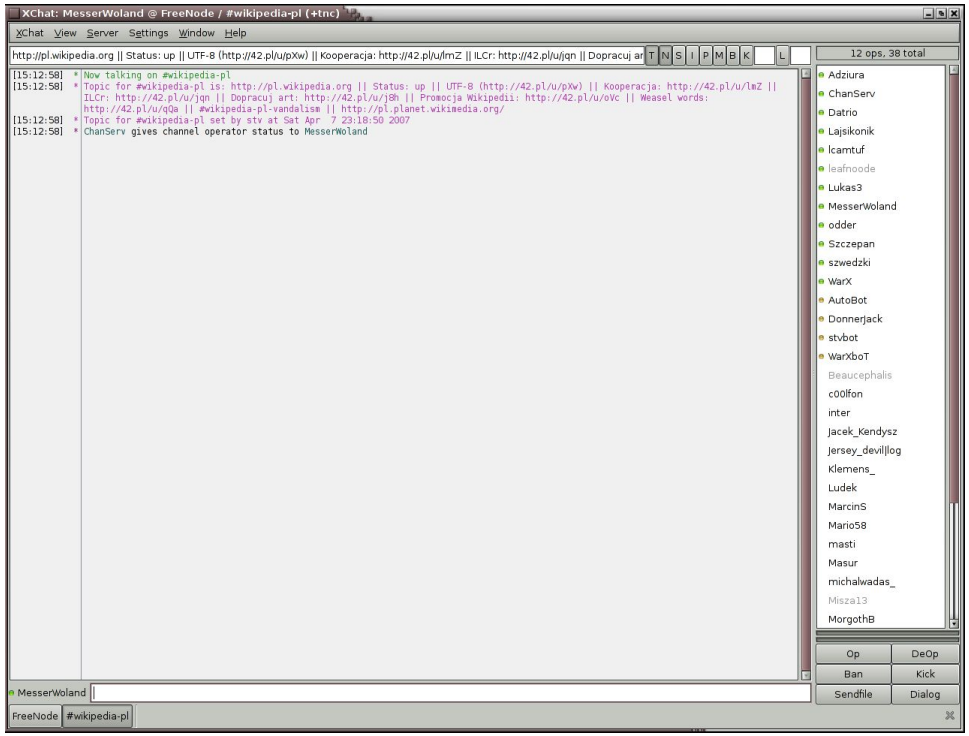
Michael Stapelberg
<michael@robustirc.net>

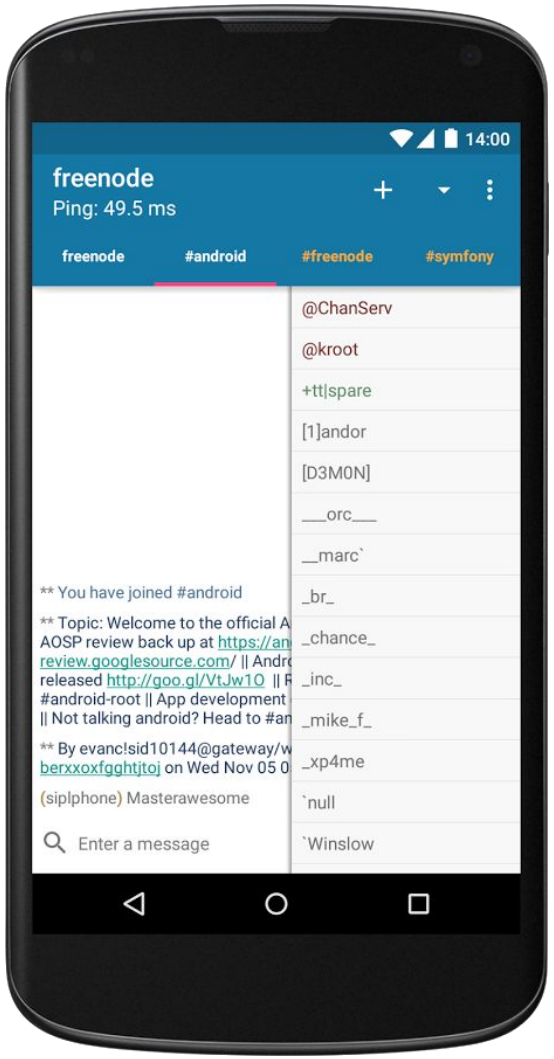
Agenda

- Motivation
- Überblick
- RobustSession-Protokoll
- Raft
- Kleingedrucktes
- Lessons learnt

Hintergrund

- IRC = Internet Relay Chat
- RFC1459, RFC281x, …





Motivation

- IRC ist in manchen Kreisen weit verbreitet
- keine überzeugende Alternative
- Stabilität verbesserungswürdig
  - TCP-Verbindungsabbrüche zerteilen das Netz (netsplit)
  - Softwareupdates, Reboots, … zerteilen das Netz
- wie Netsplits verhindern, aber kompatibel?

Idee

- TCP-Verbindungsabbrüche transparent handlen
- es gibt hochverfügbare Datenbanken, also:
- IRC-Server auf mehrere Server verteilen
  - anderes Ein-/Ausgabemodell durch Raft/Paxos

Überblick

- n RobustIRC-Server bilden 1 virtuellen IRC-Server
- Ausfälle der Minderheit (≤ floor($^{n}/_{2}$)) sind okay
  - 3 Server: 1 darf ausfallen. 5 Server: 2 dürfen ausfallen
- RobustSession-Protokoll zw. Server und Clients
- „bridge“ übersetzt IRC zu RobustSession
  - läuft auf dem selben Rechner wie der IRC-Client
  - Plugins, zukünftig vllt nativer Support in Clients?

RobustSession-Protokoll

- HTTP+JSON statt puren TCP-Verbindungen
- 4 Anfragen: POST /Create, POST, GET, DELETE
  - entspricht connect, send, receive, disconnect
- Retry für Nachrichtenversand (Duplikatfilter)
- Resume für Empfangen von Nachrichten
- Jeder Server kann jede Anfrage beantworten
  - evtl durch reverse-proxying zum Leader

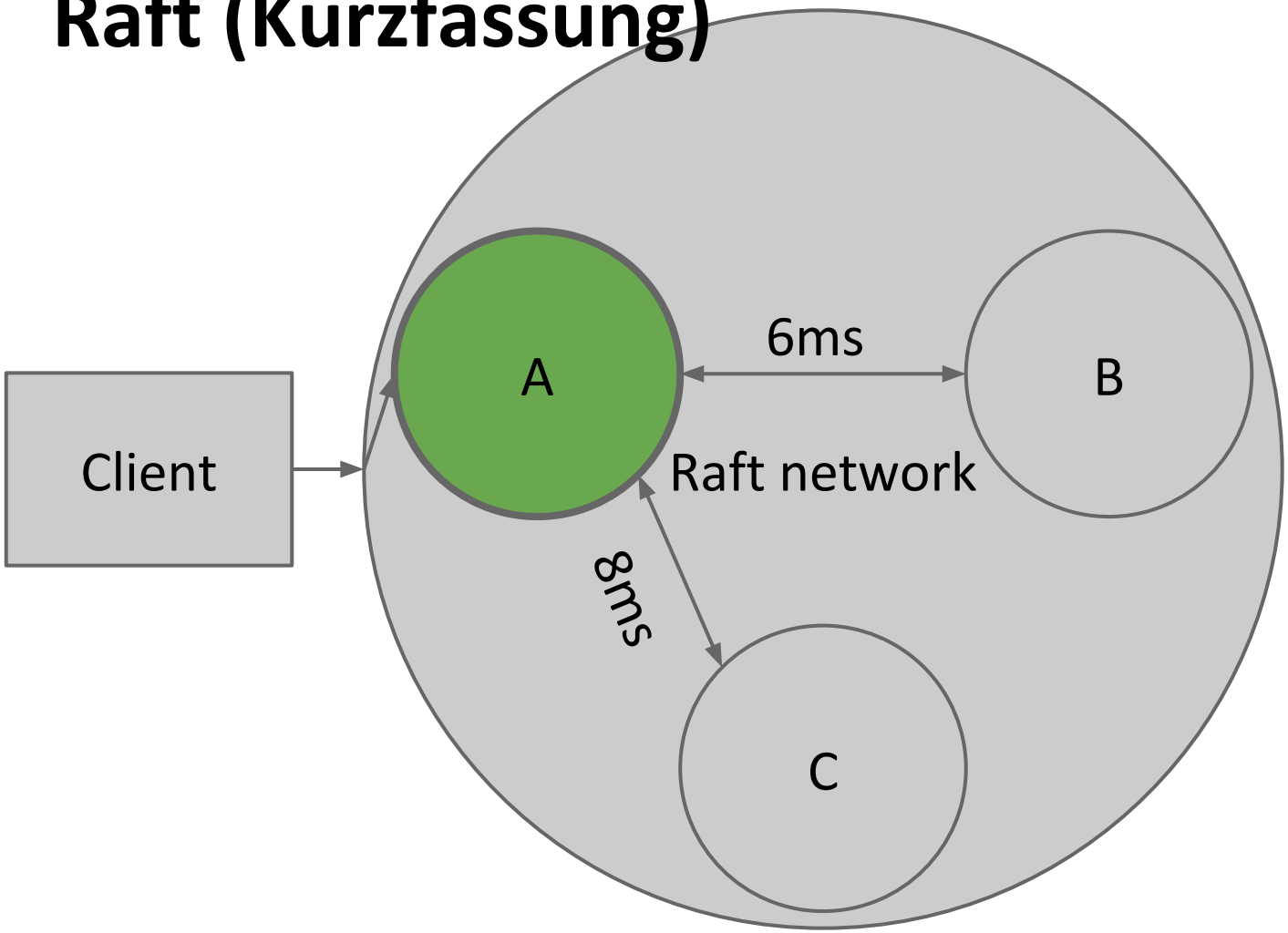
Raft

„Raft is a consensus algorithm […] It was meant to be more understandable than Paxos […], but it is also formally proven safe and offers some new features.

Raft offers a generic way to distribute a state machine across a cluster of computing systems, ensuring that each node in the cluster agrees upon the same series of state transitions.“

- http://en.wikipedia.org/wiki/Raft_(computer_science)
- http://raftconsensus.github.io/
- http://thesecretlivesofdata.com/raft/ (interaktive Visualisierung)

Raft (Kurzfassung)



Raft (Anwendung)

- IRC-Nachrichten werden via Raft repliziert
- Antworten hängen nur von Nachrichten ab

  ⟶ jeder Server generiert den selben State

  ⟶ nach Neustart wird der selbe State generiert

Kleingedrucktes

- Latenz durch Server-Verbindung limitiert
  - Latenz = Median-Latenz zw. Leader und Followern
- Ausfallsicherheit erfordert ≥ 3 Failure Domains
  - für weniger Sicherheit (z.B. nur gegen Hardwareausfälle, nicht gegen Netzausfälle) langen weniger Standorte
- Durchsatz evtl nicht hoch genug für größte Netze
  - 1000+ messages/s momentan (Proposal in Arbeit)

Demo

- localnet + server abschießen

- bei Interesse: Monitoring

Lessons learnt

- Nutzer ändern nur zögerlich ihr IRC-Setup:

  derzeit ca. 50% legacy-irc, 49% bridge, 1% plugin

- Manche Bugs sind für Laien ähnlich wie Netsplits

  z.B. Zeitabweichung nach Reboot

  (Prävention durch timesafeguard)

Fazit

- Würdet ihr RobustIRC nutzen?
- Bridge einrichten (benötigt Go):
  - `export GOPATH=~/gocode`
  - `go get -u github.com/robustirc/bridge/robustirc-bridge`
  - `$GOPATH/bin/robustirc-bridge -network=robustirc.net`
  - Im IRC-Client auf `localhost:6667` verbinden