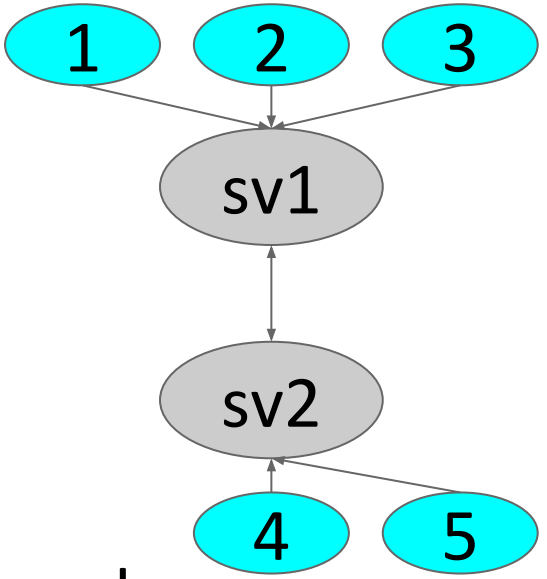
RobustIRC
or: IRC without Netsplits

GopherCon 2017

Michael Stapelberg
<michael@robustirc.net>

Motivation



- no good alternative to IRC (for us)
- our biggest problem: lack of stability
  - TCP disconnects split up an IRC network
  - hence software updates, reboots, … cause splits

    → perverse incentive to not do maintenance ☹

Idea

- server to server:

  fault-tolerant databases exist, so let's build an IRC network as a distributed system using Raft

- client to server:

  use a tunnel protocol to gloss over disconnects

Overview

- n RobustIRC servers make up 1 virtual IRC server
- minority of servers (≤ floor($^{n}/_{2}$)) can fail
  - 3 servers: 1 can fail. 5 servers: 2 can fail
- RobustSession protocol between servers/clients
- “bridge” tunnels IRC over RobustSession

Processing model

- persist incoming IRC commands using Raft
- servers are state machines

  → same state on server after process restart

  → clients can resume reading from any server

Fine print

- IRC latency ≥ median latency between all servers
  - in practice <50ms, i.e. not an issue
- truly robust networks require ≥ 3 failure domains
  - don’t place all 3 nodes in the same rack ☺

Connecting

- setting up a bridge (compiles with Go ≥ 1 (!)):
  - `go get -u github.com/robustirc/bridge/robustirc-bridge`
  - `$(go env GOPATH)/bin/robustirc-bridge -network=robustirc.net`
  - connect your IRC client to `localhost:6667`
- …or use our bridge at `legacy-irc.robustirc.net`
  - no splits on server maint., but when you disconn.

Behind the scenes: monitoring/alerting

- native Prometheus metrics
- official alerting rules file and Grafana dashboard
  - following the Google SRE book's best practices

Behind the scenes: testing with live data

- `robustirc-canary`
  - takes a snapshot (from a test or live network)
  - process with an old and new version of RobustIRC
  - generate an HTML diff of the results

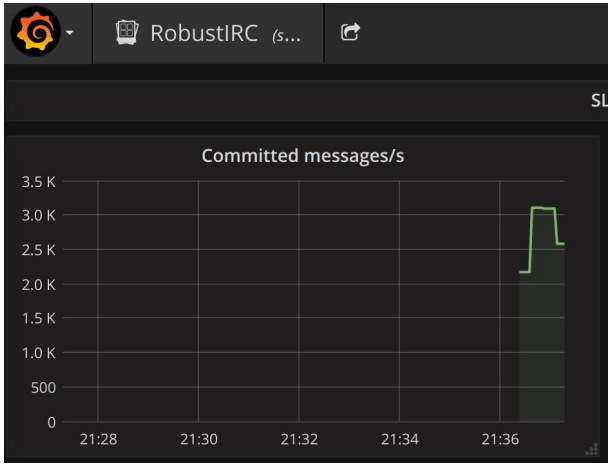
Behind the scenes: updates

- `robustirc-rollingrestart` checks health, remotely restarts 1 server at a time
- updates are pulled before starting:

```
ExecStartPre=/usr/bin/docker pull \
robustirc/robustirc:latest
```

Behind the scenes: loadtests

- robustirc-loadtest
  - starts a Kubernetes deployment
    - reproducible!
    - no own hardware needed
  - sends traffic until rates converge
  - snapshots a Grafana dashboard



Maturity

- running a production network for > 2.5 years
- sub-second startup, snapshots, …

  → can run on 3 Raspberry Pis nodes
- jepsen includes a (passing!) RobustIRC test
  - not surprising: using `hashicorp/raft` (powers consul, nomad, …)

The end

- https://robustirc.net/

  → docs/adminguide.html if you want to set it up

  → 40 minute tech talk

- please talk to me if you have questions!