Skogul – en rask titt

Kristian Lyngstøl, Februar 2020

Intro – meg og Skogul

- Jobber i Telenor, har jobbet i Redpill Linpro og Varnish Software og litt diverse
- Arrangerer The Gathering (sånn utenom pandemi)
- Elsker data og statistikk og sånt
- Skogul er tiltenkt både Telenor og TG – og flere
- Skogul er i produksjon både for Telenor og andre virksomheter
- https://github.com/telenornms/skogul

Historien

Fase en

- Du trenger å hente noe måledata og lagre det og vise det
- Du setter opp en collector, og skriver det til en database av noe slag.
- Alt er fint, alt er flott, det poller hvert femte minutt som alt annet i verden fram til nylig.

Fase to

- Løsningen er populær, du legger til mer og mer.
- Men lagringen du valgte (rrdtool, f.eks) har vist seg å være litt icky…
- Ok, du bytter ut RRD tool med noe annet, det er et større prosjekt, men du får det til. NÅ må alt være bra.

Fase 3

- Løsningen er populær. Men man vil gjerne selektivt polle noen viktige ting oftere enn andre.
- Fint om noe kunne pushe data I stedet for at alt var basert på polling.
- Hei, et nytt flott verktøy som løser dette elegant, du tar på deg et større moderniseringsprosjekt for dette er viktig.
- Det funker, NÅ er alt bra.
- Ikke sant?

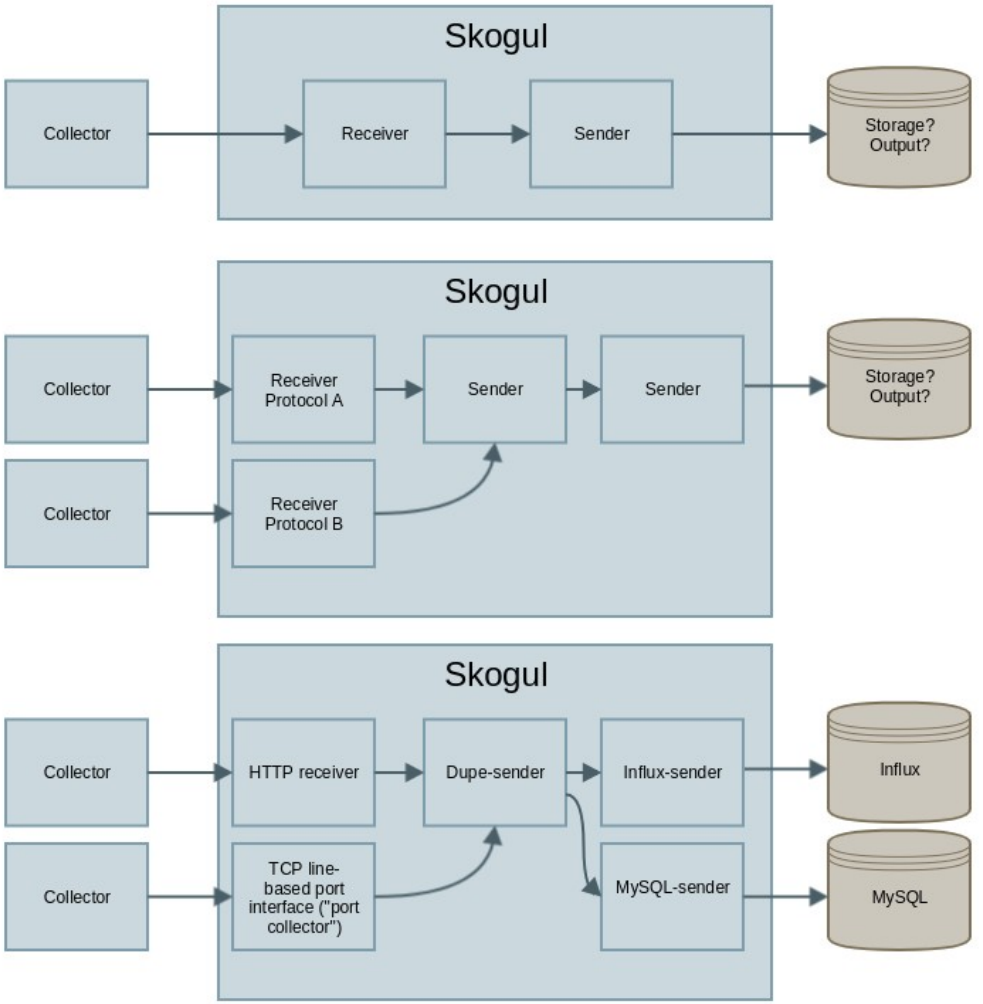
Fase 4

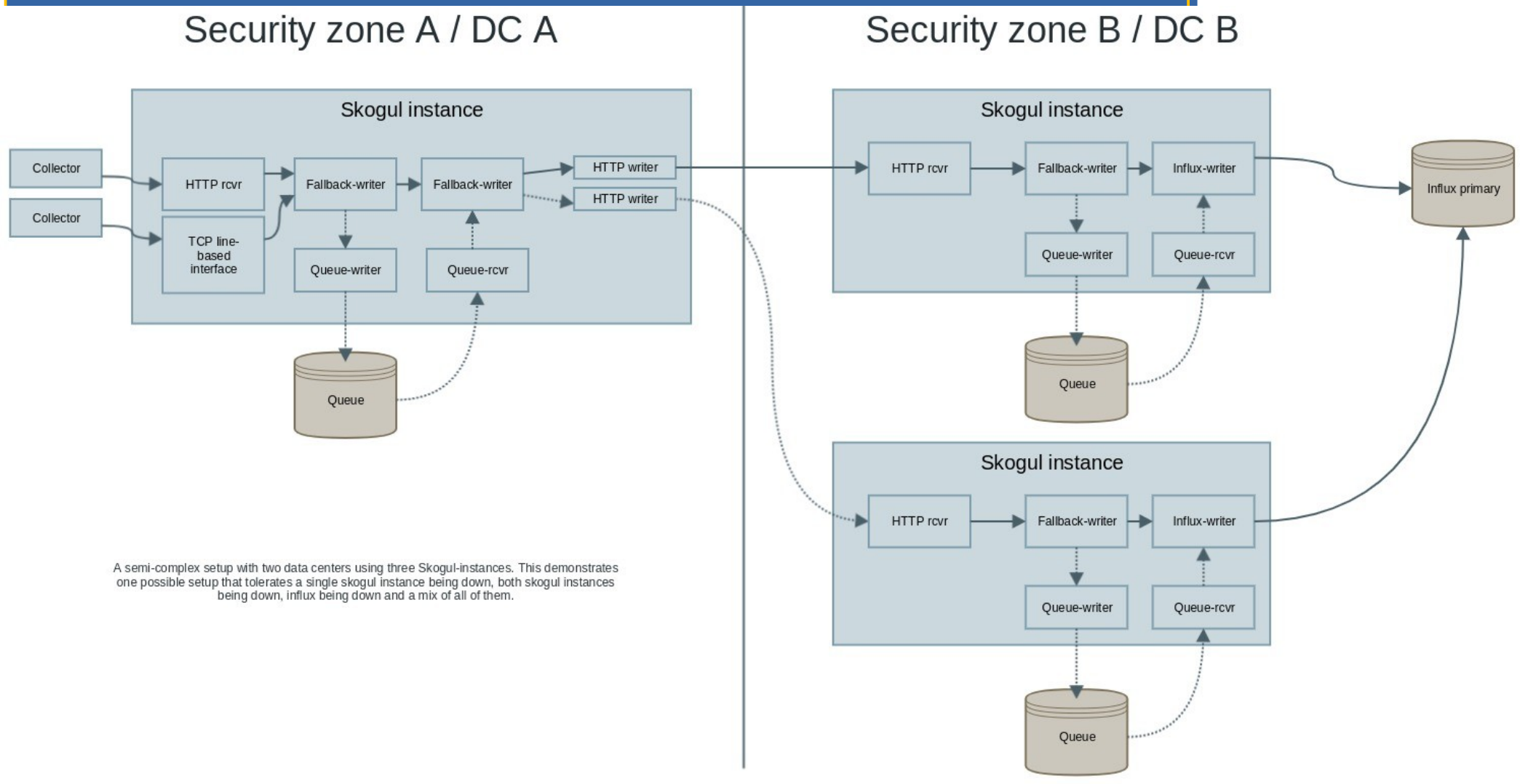
- En annen del av organisasjonen vil gjerne ha en kopi – live – av et subset av dataene dine.
- Errr…
- Og du skulle gjerne duplikert data til lab - selektivt
- Mens du fornyer hadde det også vært praktisk å skrive debug-data fra collectorene dine til fil eller kanskje postgres
- Og, err.
- ja

Skogul er pilene i arkitekturdiagrammene

Hva?

- Knøttsmå moduler
- Receivere og parsere tar i mot data og parser det til et generelt internt format
  - HTTP receiver, UDP receiver, json parser, influx line protocol parser, juniper telemetry parser, etc
- Transformere kan transformere data
  - Rename variabler, fjerne eller legge til, om-strukturere.
- Sendere pusher data videre
  - Influx, SQL, fil, stdout, splunk, mqtt, etc


Skogul
Collector
Receiver
Sender
Storage?
Output?
Skogul
Collector
Receiver
Protocol A
Sender
Sender
Storage?
Output?
Collector
Receiver
Protocol B
Skogul
Collector
HTTP receiver
Dupe-sender
Influx-sender
Influx
Collector
TCP line-
based port
interface ("port
collector")
MySQL-sender
MySQL


Security zone A / DC A
Security zone B / DC B
Skogul instance
Collector
Collector
HTTP rcvr
TCP line-based interface
Fallback-writer
Fallback-writer
HTTP writer
HTTP writer
Queue-writer
Queue-rcvr
Queue
Skogul instance
HTTP rcvr
Fallback-writer
Influx-writer
Queue-writer
Queue-rcvr
Queue
Influx primary
Skogul instance
HTTP rcvr
Fallback-writer
Influx-writer
Queue-writer
Queue-rcvr
Queue
A semi-complex setup with two data centers using three Skogul-instances. This demonstrates one possible setup that tolerates a single skogul instance being down, both skogul instances being down, influx being down and a mix of all of them.

Interndata I et nøtteskall

- Timestamp
- Metadata
  - Tenk tags, indekserte felter, identifikasjon
- Data

- Strukturen på dataen er kun begrenset av parser og sender
- Transformere brukes for å “normalisere”/falte ut data

```
{
   "timestamp": "2022-02-01T16:16:41+0200",
    "metadata": {
       "switch": "r1.noc"
    },
    "data": {
       "ae0": {
          "ifInOctets": 515,
          "ifOutOctets": 123
       },
       "ae2": {
          "ifInOctets": 525,
          "ifOutOctets": 456
       }
    }
}
```

Viktige detaljer

- ALT er moduler, og de er små og raske
- Go
- Veldig lite boilerplate I moduler – lette å skrive
- Json-config parses direkte til interne datastrukturer
- Regelmessige releaser
  - Foreløpig kun RPM og generisk binær ferdig-lagd

```go
// Backoff sender will send to Next, but retry up to Retries times, with
// exponential backoff, starting with time.Duration
type Backoff struct {
        Next    skogul.SenderRef `doc:"The sender to try"`
        Base    skogul.Duration  `doc:"Initial delay after a failure. Will double for each retry"`
        Retries uint64           `doc:"Number of retries before giving up"`
        holdoff uint64
}

// Send with a delay
func (bo *Backoff) Send(c *skogul.Container) error {
        var err error
        delay := bo.Base.Duration
        t := atomic.LoadUint64(&bo.holdoff)
        if t > 0 {
                time.Sleep(delay)
        }
        for i := uint64(1); i <= bo.Retries; i++ {
                err = bo.Next.S.Send(c)
                if err == nil {
                        if i > 1 {
                                atomic.AddUint64(&bo.holdoff, 1-i)
                        }
                        return nil
                }
                atomic.AddUint64(&bo.holdoff, 1)
                time.Sleep(delay)
                delay = delay * 2
        }
        return err
}
```

Eksmpler på bruksområder

- Motta SNMP data fra gammel poller, send til ny+gammel storage
- Duplisere data over flere DC
- Sende logg-data både til postgres og splunk
- Hente data fra en MQTT-bus
- Pushe alarm-data fra en løsning til en annen
- Motta influx-formatert data fra telegraf, kopiere det til postgres, og sende det videre til influx igjen

Push-basert telemetri fra junos

- For Juniper-svitsjer og rutere: Bytter ut SNMP med GPB-basert løsning
- Tilsvarende kommer på andre plattformer
    - Cisco sin streaming/push-telemetri er nok ikke langt unna
- Alt som kan sende JSON-data kan parses, men krever litt arbeid med config

Ytelse

- Lagd for å kunne motta data flere millioner devicer
- Vil bruke noe CPU, bruker minimalt med minne selv ved stor last, nøyaktig profil avhenger veldig av config.

Ytelse del 2

- Mottar fint data om noen millioner interfacer over tusener av devicer på en aldrende 8-core server og bruker ikke mye CPU eller minne
  - (Det parses og videresendes over https)
- Målet er at man ikke skal merke Skogul sin egen last uansett trafikk.

Ikke-mål

- Skal ikke erstatte tradisjonelle pollere/collectorer, køer, busser eller databaser – men integrere ved behov.
- Skal ikke erstatte syslog f.eks, men kan supplere

Veien videre

- Flere moduler – de lages etter behov
- Bedre debug-output (mindre), her er det gjort viktige steg nylig
- Bibliotek-config: Tenk ferdig-config for forskjellige plattformer du kan bruke fritt. Vi har for Junos, jeg har jobbet med BigIP