Switch_

Go for rsync Diversity

Simon Leinen <simon.leinen@switch.ch>, Switch (AS559)

RIPE 90 Routing WG, 15 May 2025

Management Summary

TL;DL

- I tried to use **Michael Stapelberg**[1]'s "gokrazy" rsync variant for fetching RPKI material (ROAs) with Routinator and FORT.
- It works[2]
- RRDP is still the better protocol, *ceterum censeo rsync delendum esse(???)*

[1] He did all the work. Thank you, Michael!

[2] ~~For some value of "works"~~With a bit of configuration fiddling,
reliably with acceptable performance,
and (small) manageability impact

Diversity for RPKI Robustness

- Many operators use different RPKI caches/validators, e.g. two of
  - Routinator 3000
  - rpki-client
  - FORT
- Should we use different rsync implementations as well?
  - (Assumption: We still need rsync at all until everyone has moved to RRDP)

Rsync variants

- “Tridge” rsync
- openrsync (in OpenBSD and macOS(?))
- **Gokrazy rsync**
  - Written from scratch/spec in Go, a memory-safe language
  - By Michael Stapelberg (of router7 fame)
- (possibly others, not all well supported anymore)

Why Go?

- “Memory safe”, i.e. no buffer overflows and use-after-free
  - Remember CVE-2024-12084 & friends?
- Easy multithreading model, useful for networking tools

Why “gokrazy” rsync?

- Michael has large-systems and Go “street credibility”
- He expressed willingness to help with this use case (implement missing options etc.)
- We live in the same ~~town~~ country

gokr-rsync compatibility status

- rpki-client: Assumes support for (too) many rsync options
- FORT: works, as of May 2025—see below
- Routinator 3000: works, as of March 2025

/etc/routinator/routinator.conf:

```
rsync-command = "/usr/bin/gokr-rsync"
rsync-args = ["--compress", "--no-motd", "--contimeout=10"]
```

Measurement results

FORT 1.6.6, rsync.priority=90 (prefer rsync over RRDP)

	Tridge rsync	gokr-rsync
Initial load time	~980s (16’20”)	~1600s (26’40”)
Subsequent refresh times	~823–1’001s	~751–986s
#Validated ROAs after initial load	687’868	693’054
#Validated ROAs over a few hours	687’868–693’075	692’863–693’079

Possible Future Work

- Add missing command-line args `(--include/--exclude` etc.)
- Maybe add diverse return codes like Tridge rsync
- Use as library for Go-based RPKI implementations?

BUT WHY?

- Haven’t we moved past rsync for RPKI?
- RRDP is the future! (Or we’ll come up with something else…)
- Yes, but…
  - Even when deprecated, we’ll have to support rsync for RPKI for “a while” (years)
  - Keeping old infrastructure running—not sexy but someone has to do it (to enable blockchain, AI…)
- (OK, the *actual* reason was that I wanted to learn Go from the pros… ;-)

Switch_