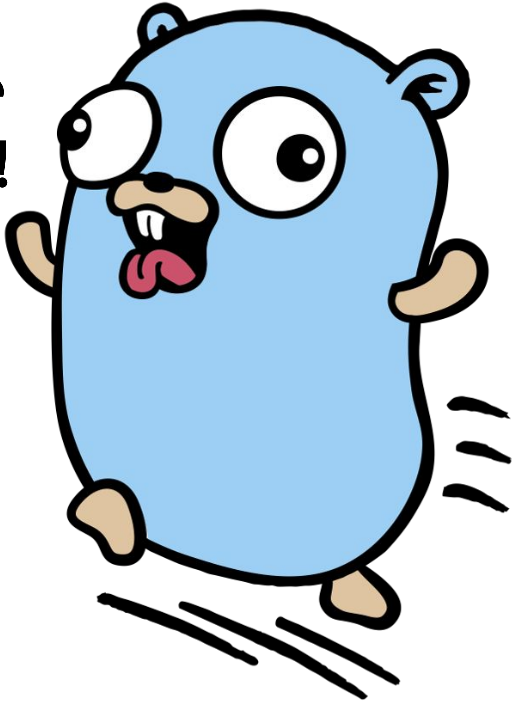
gokrazy

Build Go appliances for the Raspberry Pi using gokrazy!

Michael Stapelberg

GPN

2023-Jun-10

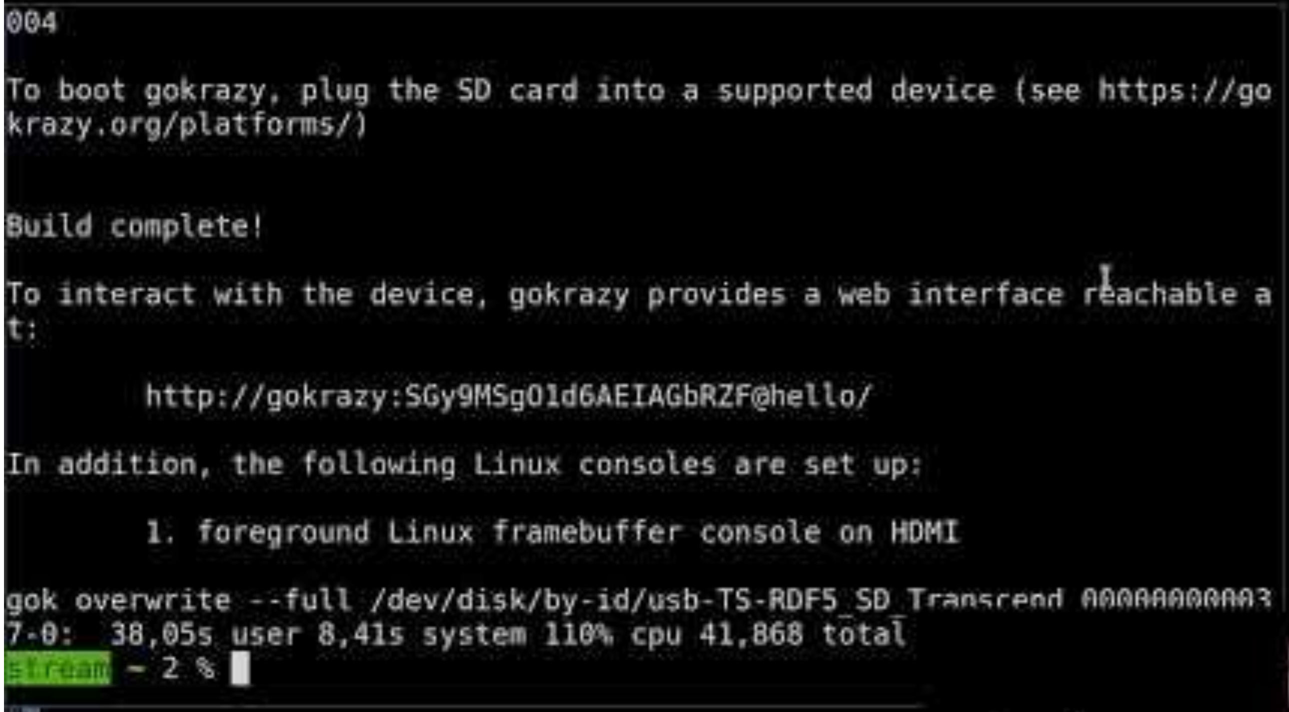


Agenda

- Introduction: what is gokrazy?
  - Demo: First Installation and Adding Programs
  - Supported Hardware
  - Fully automated Linux updates
- Notable gokrazy appliances
- Notable Go software you can run
- Building with gokrazy
- Future developments

What is gokrazy?

- Deploy your Go programs as appliances to a Raspberry Pi or PC! 🚀
- 100% written in Go — the only non-Go parts are Linux + Raspi bootloader
  - no C userland on the device! no glibc, no OpenSSL, no package managers, etc.
- Enjoy Go’s strengths, uniformly for your entire system
  - all components managed as Go modules (modify any part locally with the replace directive)
  - very quick compilation times (great for interactive development!)



```
004

To boot gokrazy, plug the SD card into a supported device (see https://go
krazy.org/platforms/)


Build complete!

To interact with the device, gokrazy provides a web interface reachable a
t:

        http://gokrazy:SGy9MSgO1d6AEIAGbRZF@hello/

In addition, the following Linux consoles are set up:

        1. foreground Linux framebuffer console on HDMI

gok overwrite --full /dev/disk/by-id/usb-TS-RDF5_SD_Transcend_00000000003
7-0:  38,05s user 8,41s system 110% cpu 41,868 total
stream ~ 2 %
```

gok overwrite

Raspberry Pi HDMI

Technical level: on disk

- partition 1: bootloader files (Pi), EFI ESP (PC)
- partition 2+3: root file system (A/B update scheme)

  read-only, compressed SquashFS image
- partition 4: permanent data partition (ext4)

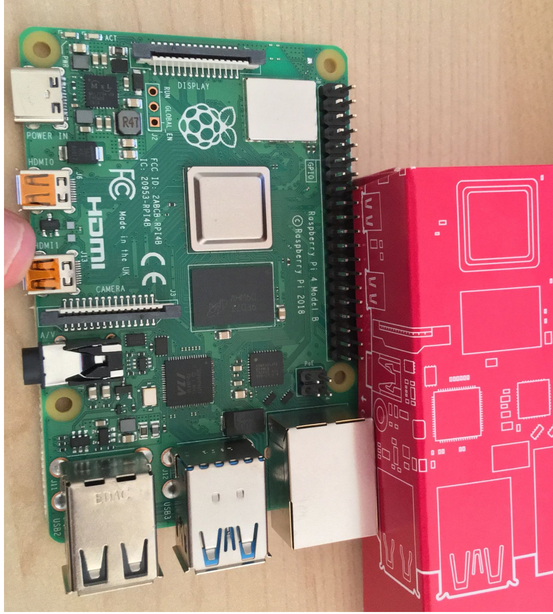
  not strictly necessary for stateless appliances :)

#	size	name	FS
1	100 MB	boot	FAT
2	500 MB	rootA	Squash
3	500 MB	rootB	Squash
4	Rest	perm	ext4

Technical level: at runtime

- gokrazy init system
  - supervises installed programs, displays the web interface, (sends logs to syslog server)
  - provides network update functionality
- gokrazy DHCP and NTP clients
- For interactive debugging, log into `breakglass` (gokrazy’s SSH server)
  - default environment: busybox (embedded Linux toolset)
  - Bring Your Own Software (e.g. strace, tcpdump, …)

Supported Hardware



- Raspberry Pi 3, Pi 4, Pi Zero 2W

  (first arm64 Pis that are supported by upstream Linux)

- Standard PCs (amd64, UEFI or MBR)

  example: PC Engines APU, Proxmox

- Community-supported alternatives:

  Raspberry Pi, Pi 2, Pi Zero W (with Raspberry Pi Linux kernel fork)

  Odroid XU4, HC1, HC2 (custom kernel)

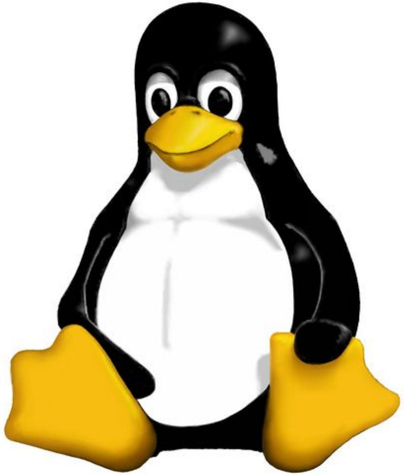
Supported Hardware: WiFi and Bluetooth

- Encrypted WiFi networks work on the Raspi (since March 2022 🎉)

```
echo '{"ssid": "I/O Tee", "psk": "secret"}' > /perm/wifi.json
```

- Bluetooth Low Energy (BLE), for example for IOT sensors
    - But no full Bluetooth stack (e.g. for audio, or wireless keyboards)

Automated Linux updates

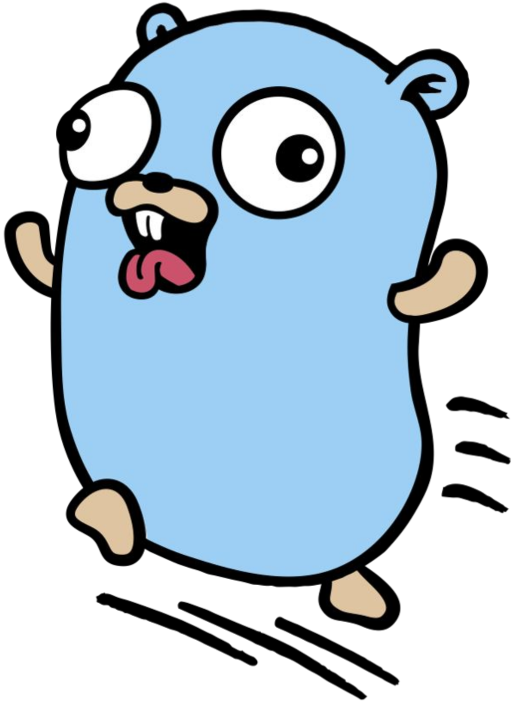


- Goal: make new Linux releases available ASAP

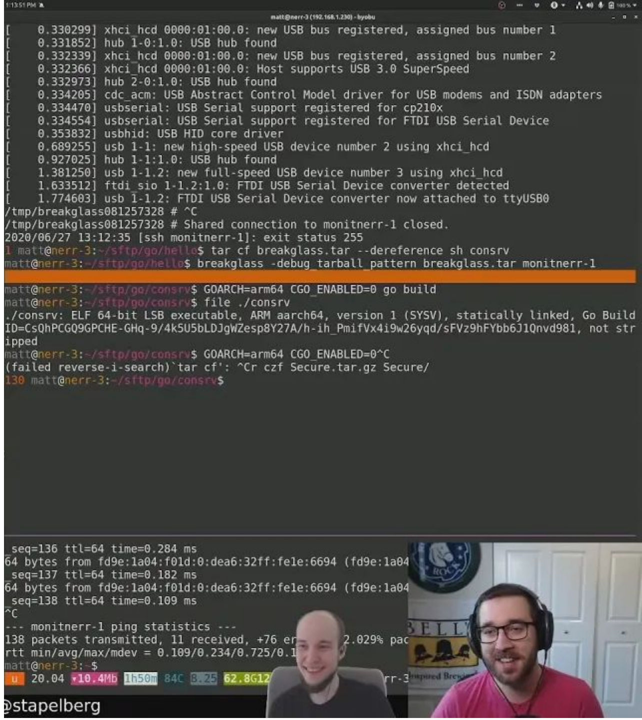
- Look for new releases on kernel.org daily, submit GitHub PR
  - Automation builds new kernel on GitHub Actions, amends the PR
  - Automation deploys new gokrazy builds on “sacrificial” Raspberry Pis
  - If the devices boot and pass the testsuite successfully, the PR gets merged!

- Track record: most versions in < 24h, with issues typically in a few days

Notable gokrazy appliances



Example appliance: consrv



```
[    0.330299] xhci_hcd 0000:01:00.0: new USB bus registered, assigned bus number 1
[    0.331852] hub 1-0:1.0: USB hub found
[    0.332339] xhci_hcd 0000:01:00.0: new USB bus registered, assigned bus number 2
[    0.332366] xhci_hcd 0000:01:00.0: Host supports USB 3.0 SuperSpeed
[    0.332973] hub 2-0:1.0: USB hub found
[    0.334205] cdc_acm: USB Abstract Control Model driver for USB modems and ISDN adapters
[    0.334470] usbserial: USB Serial support registered for cp210x
[    0.334554] usbserial: USB Serial support registered for FTDI USB Serial Device
[    0.353832] usbhid: USB HID core driver
[    0.689255] usb 1-1: new high-speed USB device number 2 using xhci_hcd
[    0.927025] hub 1-1:1.0: USB hub found
[    1.381250] usb 1-1.2: new full-speed USB device number 3 using xhci_hcd
[    1.633512] ftdi_sio 1-1.2:1.0: FTDI USB Serial Device converter detected
[    1.774603] usb 1-1.2: FTDI USB Serial Device converter now attached to ttyUSB0
/tmp/breakglass081257328 # ^C
/tmp/breakglass081257328 # Shared connection to monitnerr-1 closed.
2020/06/27 13:12:35 [ssh monitnerr-1]: exit status 255
1 matt@nerr-3:~/sftp/go/hello$ tar cf breakglass.tar --dereference sh consrv
matt@nerr-3:~/sftp/go/hello$ breakglass -debug_tarball_pattern breakglass.tar monitnerr-1

matt@nerr-3:~/sftp/go/consrv$ GOARCH=arm64 CGO_ENABLED=0 go build
matt@nerr-3:~/sftp/go/consrv$ file ./consrv
./consrv: ELF 64-bit LSB executable, ARM aarch64, version 1 (SYSV), statically linked, Go Build
ID=CsQhPCGQ9GPCHE-GHq-9/4k5U5bLDJgWZesp8Y27A/h-ih_PmifVx4i9w26yqd/sFVz9hFYbb6J1Qnvd981, not str
ipped
matt@nerr-3:~/sftp/go/consrv$ GOARCH=arm64 CGO_ENABLED=0^C
(failed reverse-i-search)`tar cf': ^Cr czf Secure.tar.gz Secure/
130 matt@nerr-3:~/sftp/go/consrv$
```

```
_seq=136 ttl=64 time=0.284 ms
64 bytes from fd9e:1a04:f01d:0:dea6:32ff:fe1e:6694 (fd9e:1a04
_seq=137 ttl=64 time=0.182 ms
64 bytes from fd9e:1a04:f01d:0:dea6:32ff:fe1e:6694 (fd9e:1a04
_seq=138 ttl=64 time=0.109 ms
^C
--- monitnerr-1 ping statistics ---
138 packets transmitted, 11 received, +76 er        2.029% pac
rtt min/avg/max/mdev = 0.109/0.234/0.725/0.1
matt@nerr-3:~$
 u  20.04 ▾10.4Mb 1h50m 84C 8.25 62.8G12                   rr-3
```

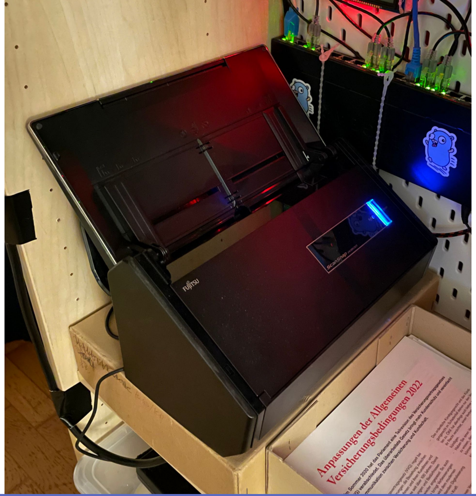
- Do you have a homelab or small rack at work?
- Serial console (RS232) access can be handy, e.g. when your server doesn’t come online!
- consrv makes serial ports available over SSH (watch Matt Layher and me start the project on stream)

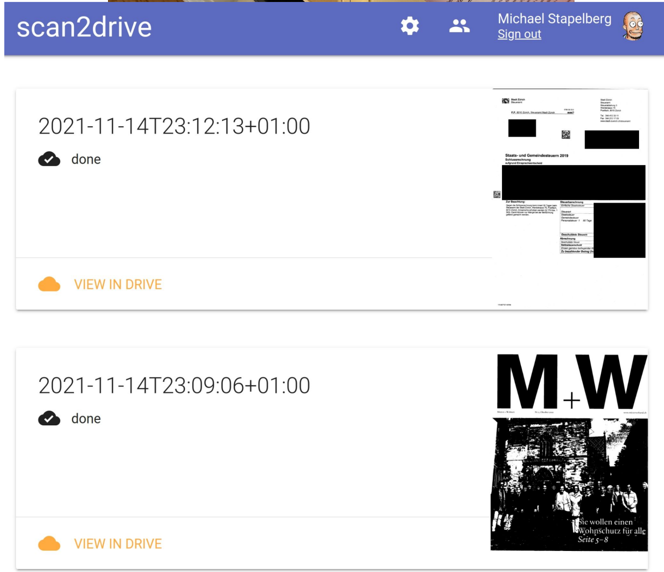
Example appliance: scan2drive

- Do you ever get paper mail?

  Are you diligently scanning and organizing it?

- scan2drive makes it easy to scan to Google Drive
  - converts scans into black & white PDFs
  - full-text search on Google Drive!
  - supports the Fujitsu ScanSnap iX500 (duplex)
  - supports any AirScan-compatible device!





scan2drive

Michael Stapelberg
Sign out

2021-11-14T23:12:13+01:00

done

VIEW IN DRIVE

2021-11-14T23:09:06+01:00

done

VIEW IN DRIVE

Example appliance: router7

- I used to use OpenWrt on a Turris Omnia

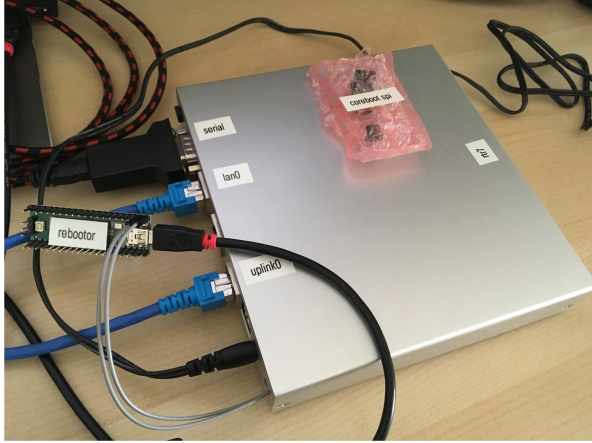
  In May 2018, an odhcp6c update broke my IPv6 😢

  I decide it would be fun to build my own router!

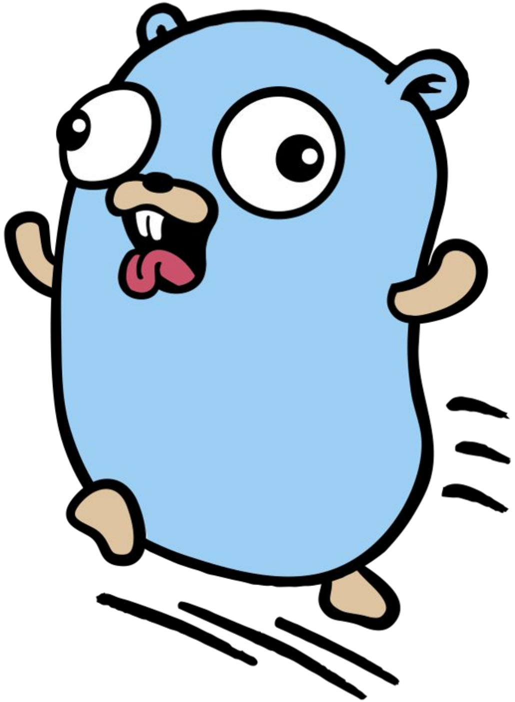
- router7 is a Go DHCP client, server, DNS forwarder, ..

  Goals: maximize connectivity, reliable, debuggable

- Built for PC Engines APU, later upgraded for 25 Gbit



Notable Go software you can run



Notable software: Prometheus

- Prometheus is a monitoring system & time series database

- Prometheus Node Exporter allows exporting metrics
  allows monitoring your Pi’s CPU, RAM, disk, …

- Prometheus Blackbox Exporter allows probing targets

- Prometheus itself works, too! (But use good durable storage, not SD cards)

Notable software: Tailscale

- Tailscale is “a secure network that just works”
  a zero-config VPN, using WireGuard in a mesh

- Make your Pi’s services available over Tailscale!

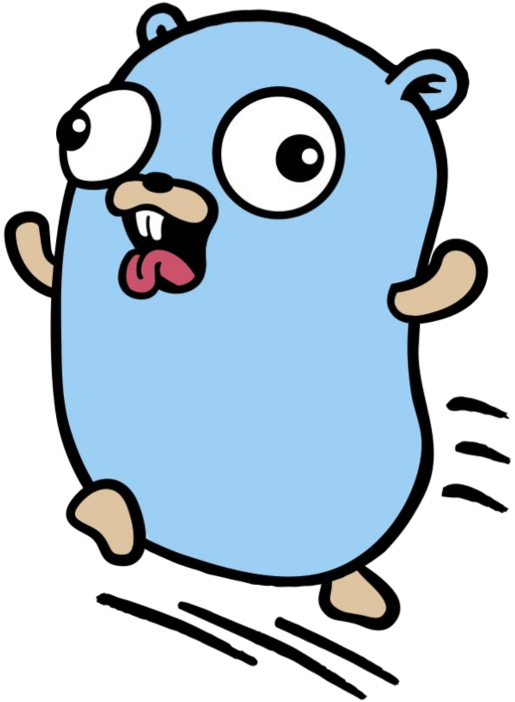
- Tailscale “Subnet Router”: make your entire LAN available 💯

Notable software: Docker containers

- Podman can run Docker containers
  available as a standalone static build
  available for gokrazy since April 2022 🎉

- Escape hatch for running non-Go programs
  - Ubiquiti UniFi controller to manage your WiFi Access Point
  - IRC chat setup: irssi in screen, with Perl scripts

Building with gokrazy



Building with gokrazy

- You supply: a runnable Go program
  - customize command-line flags, environment variables, etc.
  - details at https://gokrazy.org/development/process-interface/

- Network works out of the box — great for IOT use-cases
  - e.g., install `github.com/fhmq/hmq` to get a working MQTT server

- Hardware support: what’s included in (upstream) Linux
  - e.g., use periph.io for GPIO access: https://gokrazy.org/development/gpio/

Demo: adding a new Go package from scratch

- Live demo:
  - `gok -i gpn new; cd ~/gokrazy/gpn`
  - `mkdir hey; cd hey`
  - `go mod init hey`
  - `echo -e "package main\nfunc main() {}" > hey.go`
  - `gok -i gpn add .`

Interactive development: gok run

- Full update: `gok -i scanner update`, takes about 45 seconds on Pi 4

-
  ```
  cd ~/scan2drive/cmd/scan2drive
  gok -i scanner run
  ```

- (cross-)compiles, uploads into RAM, restarts updated version

No C? libjpeg-turbo counter-example

- Go’s `image/jpeg` is a nice and readable JPEG implementation

  …but libjpeg-turbo contains an Arm Neon-optimized version (SIMD)

- gokrazy is 100% Go, no C userland (no glibc)

  …but (some) C libraries can be used if statically linked

- build with `CC=aarch64-linux-gnu-gcc` env var and build flags

  `-ldflags=-linkmode external -extldflags -static`

Escape hatch: prototyping with C programs

- Not all C software can be linked statically,

  e.g. `tc` dynamically loads traffic shaping plugins

- One option is running in Docker containers (via podman)

- Another option is to copy `.so` dependencies:

  https://gokrazy.org/development/non-go/

  - No security updates! But useful for helpers like `mkfs.ext4` or similar

Future developments 🔮

- **GUS: gokrazy Update Service**

  asynchronous updates for intermittently online devices

  one-to-many updates (one image, many devices)

Thank you for your attention!

- More infos at gokrazy.org
- Questions? Talk to me after the presentations :)
- Give me feedback on this presentation!