'25

NixOS @ MSF-OCB

Ian Sollars (MSF, DevOps TL) / Sohel Sarder (MSF, EMR expert & eHealth DevOps Engineer) / Ramses de Norre (Numtide)

Why we use it and how

Introducing

MSF

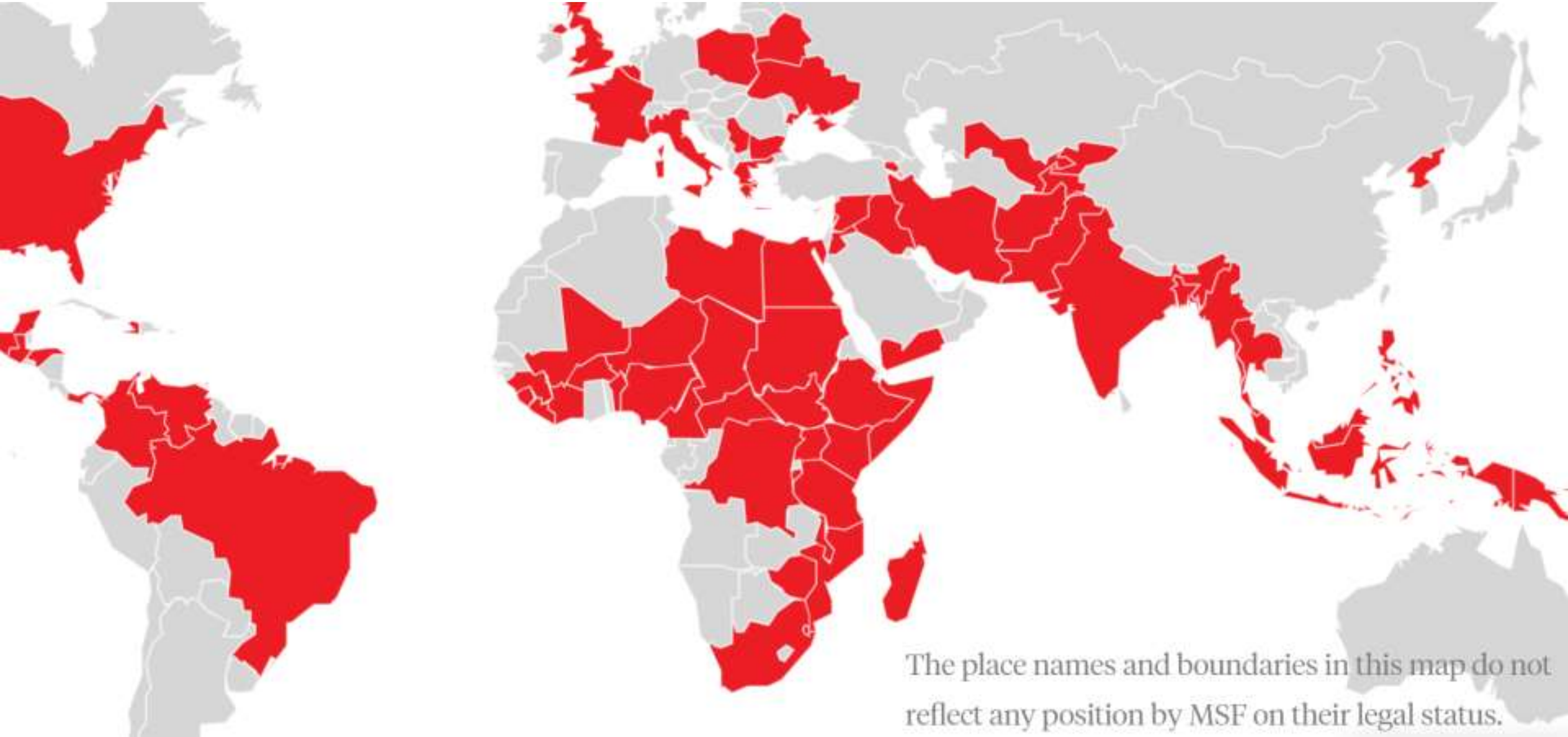
An international, <u>**independent**</u> medical humanitarian organisation

Médecins Sans Frontières (MSF) translates to ‘doctors without borders’. We provide medical assistance to people affected by conflict, epidemics, disasters, or exclusion from healthcare.

Our teams are made up of tens of thousands of professionals working in health and medical care, logistics, administration, communications, skilled trades – all bound together by our charter and serving people in need. Our actions are guided by medical ethics and the principles of **impartiality**, **independence**, and **neutrality**. We are a non-profit, self-governed, member-based organisation.

msf.org/who-we-are

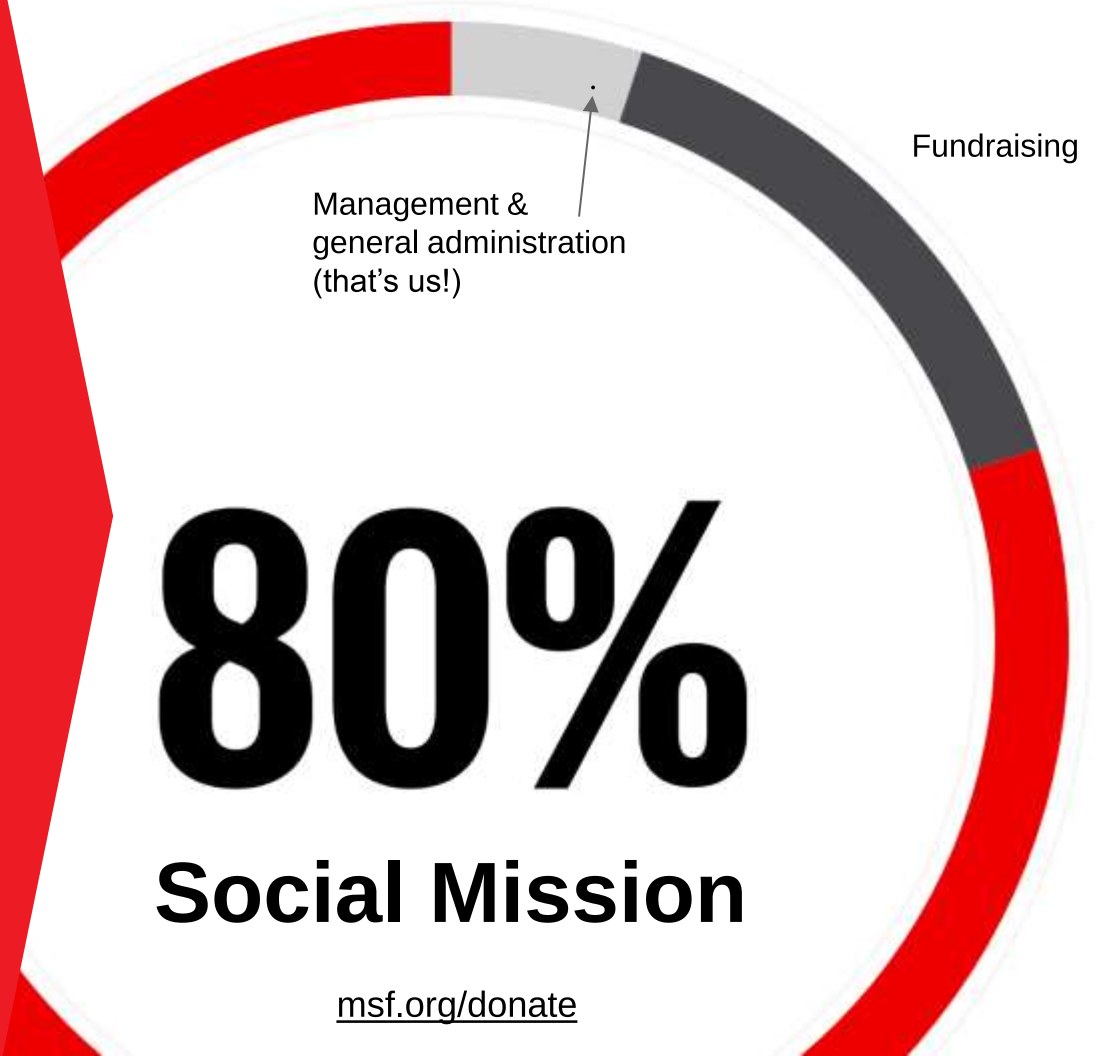
Where we work


The place names and boundaries in this map do not reflect any position by MSF on their legal status.

Funding

In 2023, 98% of our income came from 7.3 million private donors.

This is what ensures our independence.



msf.org/donate

Source: 2023 International Financial Report

16,459,000
outpatient consultations

3,724,500
malaria cases treated

3,295,700
vaccinations against measles in response to an outbreak

1,946,300
emergency room admissions

1,368,700
patients admitted

499,500
admissions of malnourished children to outpatient feeding programmes

493,900
individual mental health consultations

462,200
families received distributions of relief items

337,000
births assisted, including caesarean sections

161,000
severely malnourished children admitted to inpatient feeding programmes

125,900
surgical interventions involving the incision, excision, manipulation or suturing of tissue, requiring anaesthesia

70,600
patients treated for cholera

62,200
people treated for sexual violence

44,500
people receiving HIV antiretroviral treatment

23,000
people with advanced HIV under MSF care

22,700
people started on first-line tuberculosis treatment

5,810
people started on hepatitis C treatment

4,650
people rescued at sea

Violent clashes in the central neighbourhood of Bel Air in Port-au-Prince on the 28 of February provoked the arrival of 92 patients with bullet wounds at the MSF Emergency Center of Turgeau in the space of a week. © Alexandre Marcou/MSF

MEDECINS SANS FRONTIERES

MSF is 69,000+ people (89% social mission)

OCB ‘HQ’ is ≈ 980 people in 13 countries supporting ≈ 15% of global staff

IT@OCB-HQ is ≈ 55 people in 4 places 🇧🇪 🇨🇩 🇰🇪 🇱🇧

5 people in IT@OCB-HQ work operationally with NixOS. **(None 100%)**

Project support & consultancy from Numtide!

MEDECINS SANS FRONTIERES

Source: 2023 MSF Staff Data & Trends Report (internal)

What on-prem & cloud infra do we use?

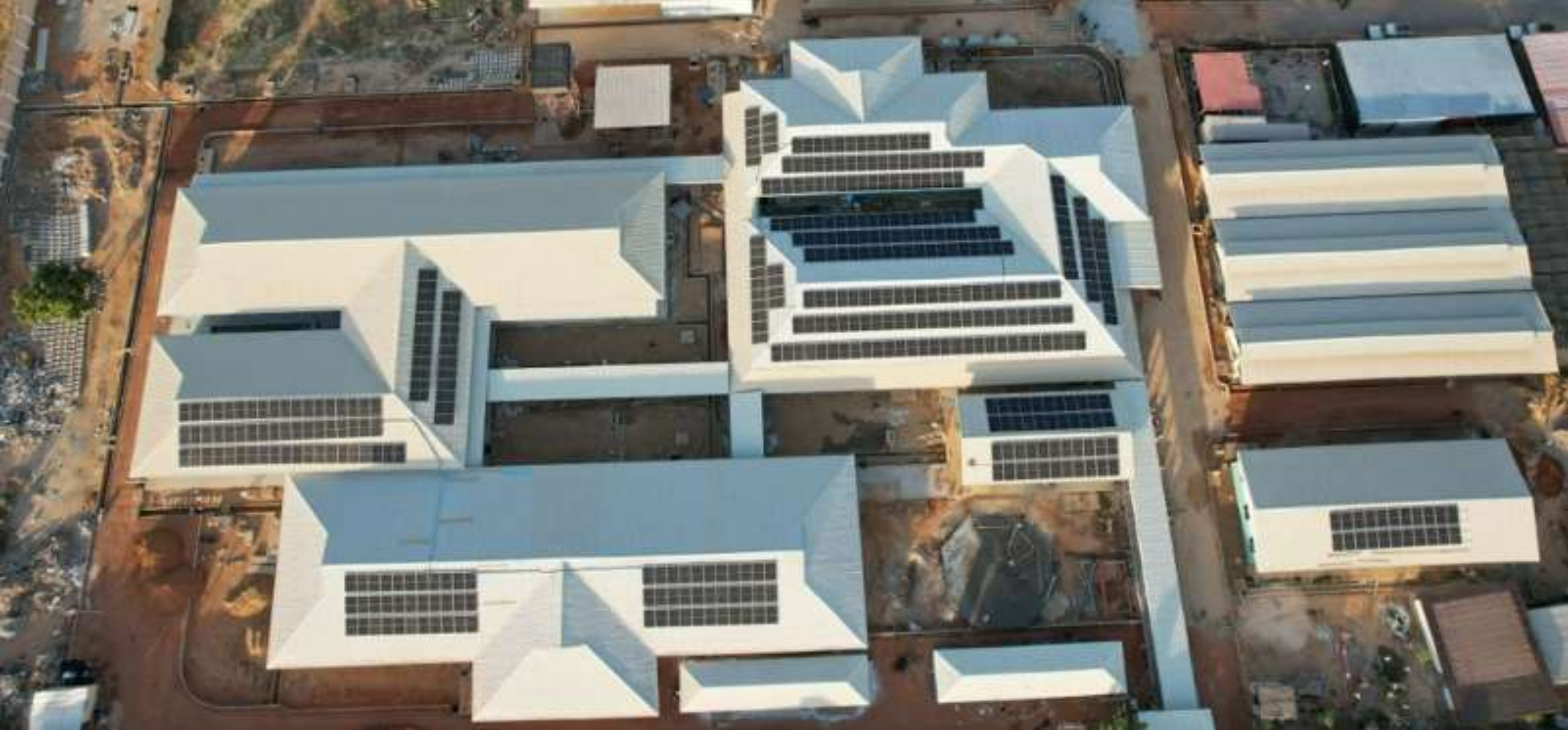
The mention of following products and technologies is for informational purposes only. MSF does not endorse, promote, or recommend any specific product, service, or technology mentioned here. References to any company, product, or service should not be construed as an endorsement by MSF.

- Over 100 servers/VMs as of 7/24
- Operating in cloud & resource-constrained/disconnected environments across 20+ countries.
- **Patient data stays in the patient's country:** that's a big reason for the dispersal.

aws

vmware® by Broadcom

MEDECINS SANS FRONTIERES


Nelifa Keji Hospital Maiduguri, Nigeria, © Yusuf Anjikwi Mshelia/MSF

What on-site infra do we use?

- Field Network Kits (FNKs) have router & firewall, UPS, VM hosting
- Fanless industrial NUCs fit in a backpack

MEDECINS SANS FRONTIERES

A view of one of the wards of the ITFC at Nilefa Kiji nutrition hospital run by MSF in Maiduguri, Borno State in Nigeria. © Nasir Ghafoor/MSF

IT @ MSF-OCB

Alex working on the MY Bourbon Argos for SaR operations in Augusta. © Alessandro Penso/MAPS

Jean Liyolongo works late into the night at an MSF base, Monga, in Bas-Uele Province, Democratic Republic of Congo. © Diana Zeyneb Alhindawi

MEDECINS SANS FRONTIERES

What are the platform components?

The mention of following products and technologies is for informational purposes only. MSF does not endorse, promote, or recommend any specific product, service, or technology mentioned here. References to any company, product, or service should not be construed as an endorsement by MSF.

What applications do we run?

The mention of following products and technologies is for informational purposes only. MSF does not endorse, promote, or recommend any specific product, service, or technology mentioned here. References to any company, product, or service should not be construed as an endorsement by MSF.

dhis2

ORTHANC

MySQL™

Bahmni™

KEYCLOAK *

* Coming soon

Why do we use NixOS?

- We need to deploy resilient systems for critical applications.
- These systems need to evolve quickly with minimal maintenance.
- We need unified field and HQ operations.

➔ NixOS's declarative IaC approach works here.

Design Goals

Automated testing & deployment of applications, updates & security patches

Centralized & secure configuration management

Remote access with minimal network dependencies

Prevent configuration drift

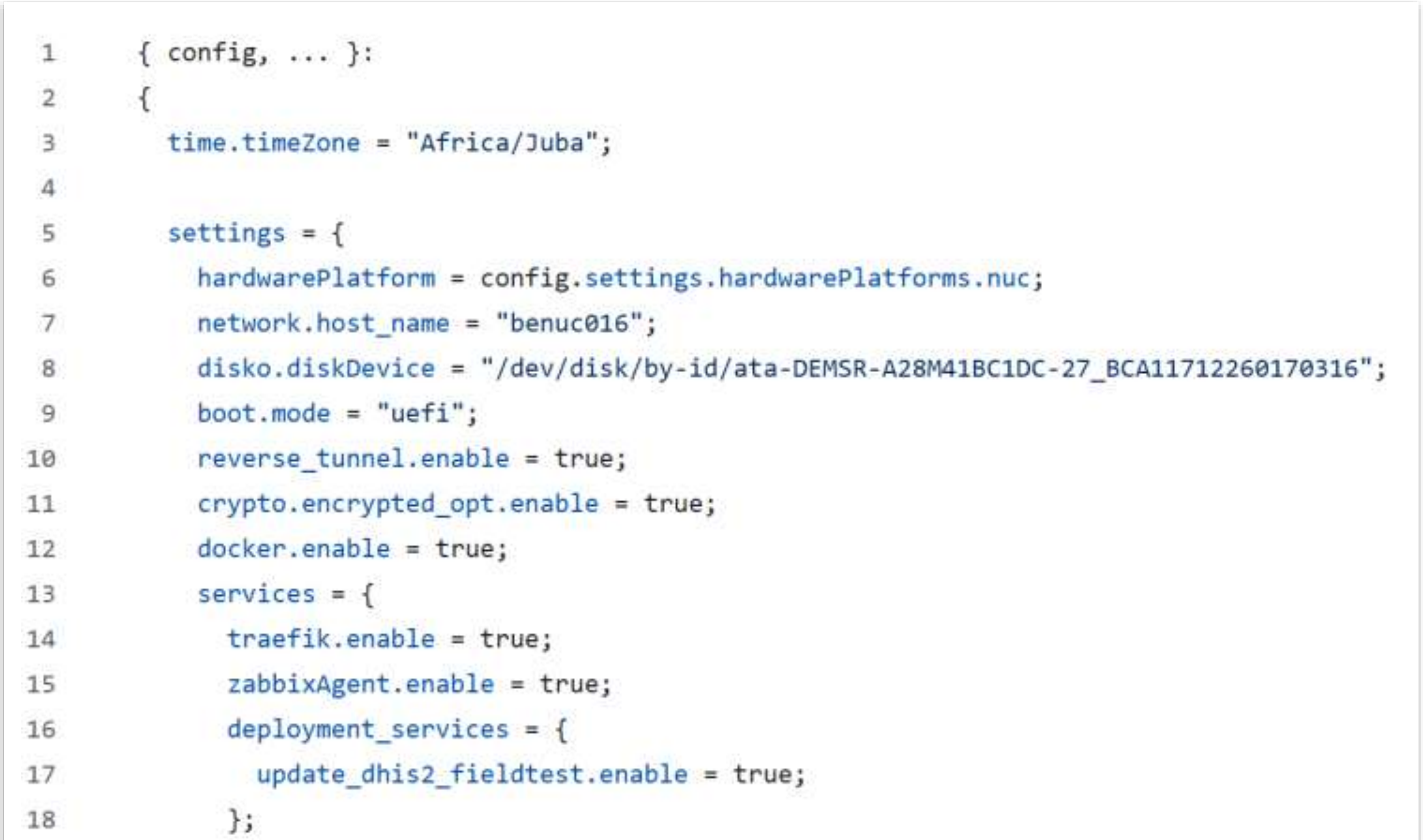
Containerized application deployments

MEDECINS SANS FRONTIERES

How do we use NixOS?

- MSF-OCB started to use a custom-made NixOS platform for the management of a fleet of Linux servers in 2018.
- We defined our servers using Nix & store the config in SCM (Git).
- The servers have a scheduled service that pulls the code & rebuilds to get updates & upgrades.

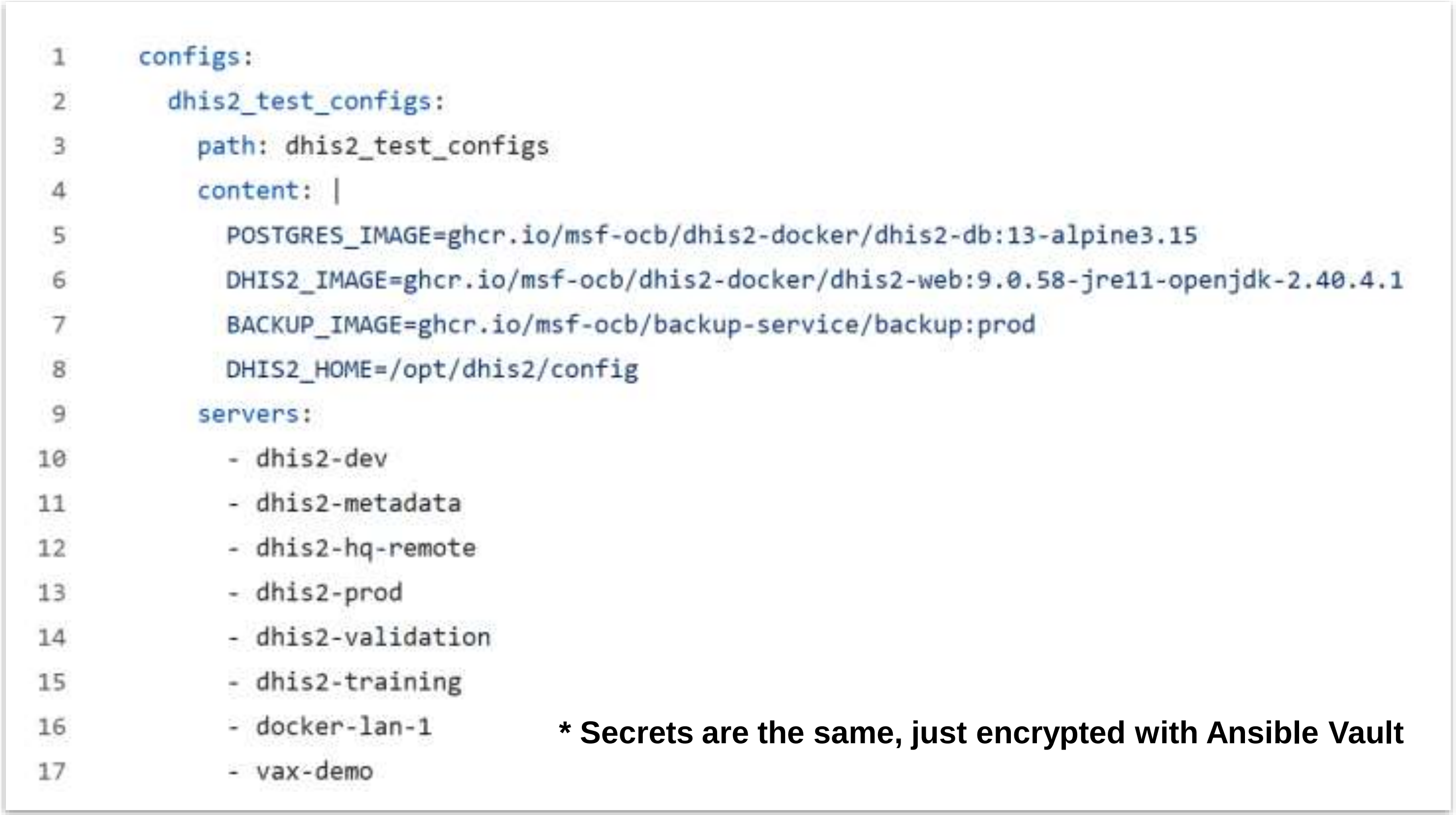
Centralized config management: servers



```
{ config, ... }:
{
  time.timeZone = "Africa/Juba";

  settings = {
    hardwarePlatform = config.settings.hardwarePlatforms.nuc;
    network.host_name = "benuc016";
    disko.diskDevice = "/dev/disk/by-id/ata-DEMSR-A28M41BC1DC-27_BCA11712260170316";
    boot.mode = "uefi";
    reverse_tunnel.enable = true;
    crypto.encrypted_opt.enable = true;
    docker.enable = true;
    services = {
      traefik.enable = true;
      zabbixAgent.enable = true;
      deployment_services = {
        update_dhis2_fieldtest.enable = true;
      };
```

Centralized config management: YAML*



```yaml
configs:
  dhis2_test_configs:
    path: dhis2_test_configs
    content: |
      POSTGRES_IMAGE=ghcr.io/msf-ocb/dhis2-docker/dhis2-db:13-alpine3.15
      DHIS2_IMAGE=ghcr.io/msf-ocb/dhis2-docker/dhis2-web:9.0.58-jre11-openjdk-2.40.4.1
      BACKUP_IMAGE=ghcr.io/msf-ocb/backup-service/backup:prod
      DHIS2_HOME=/opt/dhis2/config
    servers:
      - dhis2-dev
      - dhis2-metadata
      - dhis2-hq-remote
      - dhis2-prod
      - dhis2-validation
      - dhis2-training
      - docker-lan-1
      - vax-demo
```

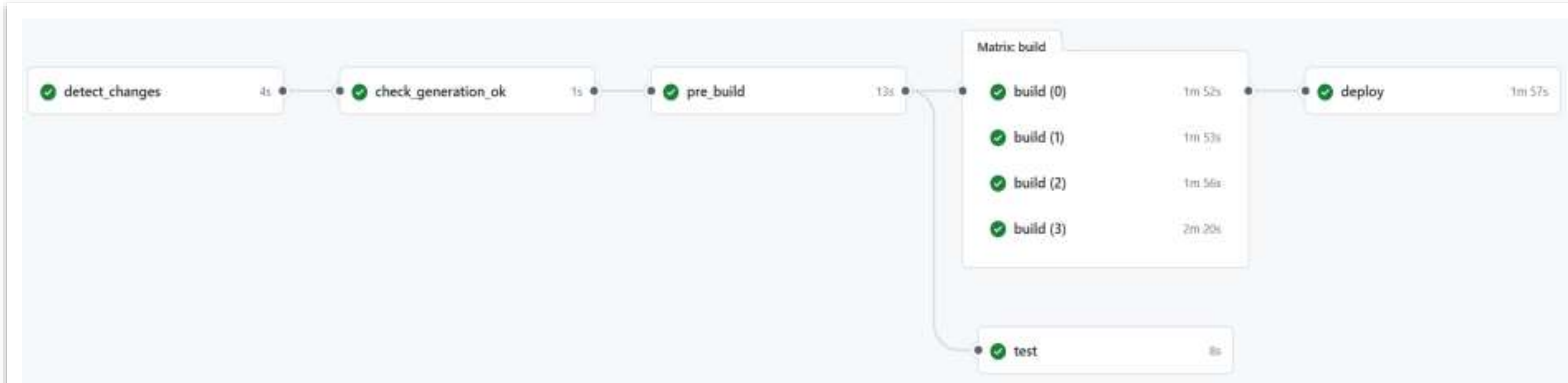
*** Secrets are the same, just encrypted with Ansible Vault**

Shelters seen from a hilltop in Jamtoli refugee camp, Cox’s Bazar, Bangladesh. © Saikat Mojumder/MSF

Automated testing & deployment

The declarative nature of NixOS has made several design goals easier.

- We manage server configuration and deployment process centrally via GitHub repos.
- Configuration changes are deployed automatically.
- We use GitOps for change reviews and tracking.
- We use GitHub Actions for CI/CD.



Automated patching: patches & version updates

We do a NixOS version upgrade twice a year and run an automatic software/security patching once every week.

- We use **Nix Flakes** for maintaining our NixOS project and managing its upstream dependencies.
- Weekly flake lock bumps for security patches and updates (1-click of auto-generated PR & some sanity checks).
- Semi-automated upgrade waves (first, middle and final wave in an upgrade cycle, 3-line PR 2x/year) to keep servers up-to-date.

MEDECINS SANS FRONTIERES

Upgrade waves – what hosts go where

First wave: relays & dev machines

24.11

Middle wave – UAT, test servers, low-SLA production servers

24.11

Final wave – servers hosting mission-critical applications

24.05*

* Upgrading this Tuesday!

A picture of Sohel's desk – first wave machines. © MSF

Last wave!

Gregor Schmeiser, an orthopaedic surgeon in the MSF Kunduz Trauma Centre in Afghanistan, prepares for surgery. © Nava Jamshidi

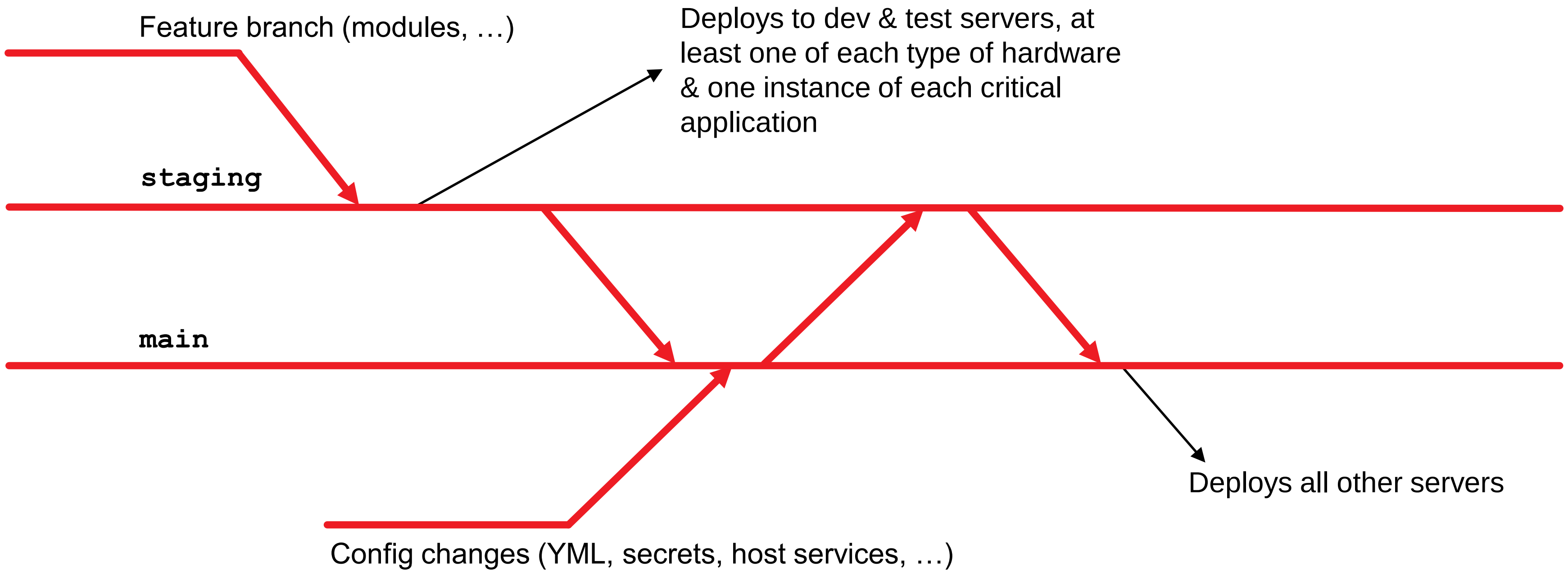
How do we test our Nix code?

The same way you do 😊

- Build NIX closures in GitHub actions, with our own build server & a persistent store.
- (New) VM-based tests, same as Nixpkgs.
- Custom and major critical changes in Nix code runs on **`staging`** hosts pulling from the **`staging`** branch code before merging the code into **`main`** branch.

This minimizes the disturbance to our operations and decouples development & deployment.

Staging & production

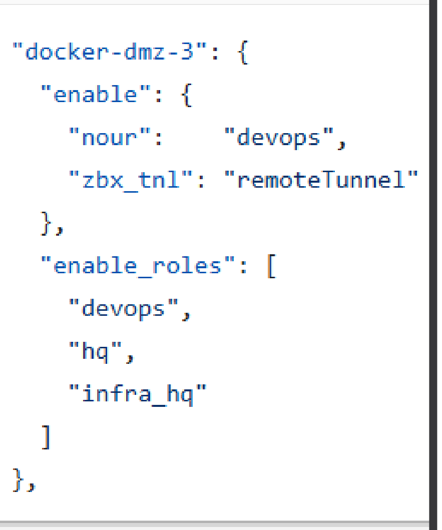

Feature branch (modules, …)
Deploys to dev & test servers, at least one of each type of hardware & one instance of each critical application
staging
main
Deploys all other servers
Config changes (YML, secrets, host services, …)

How do we manage servers?

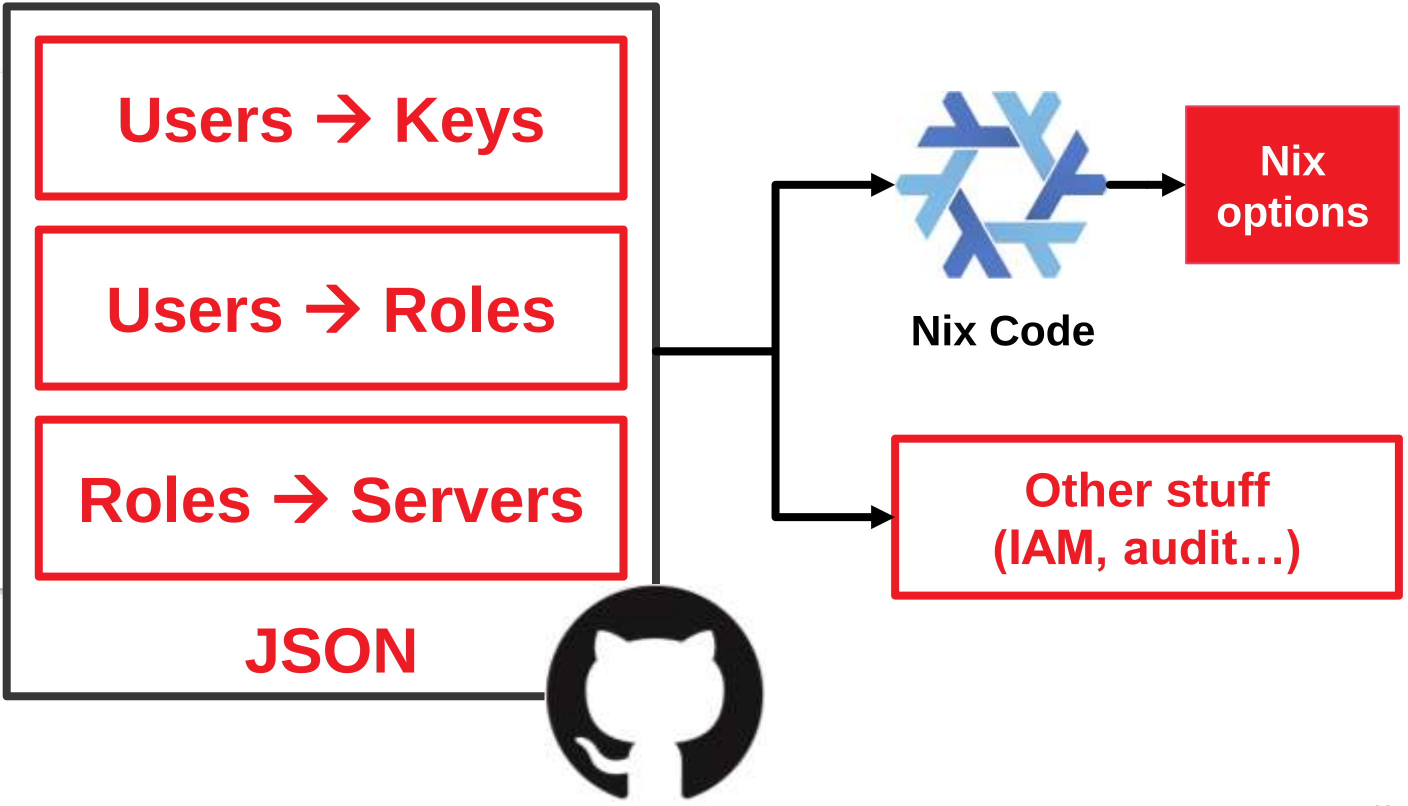
Remote access with minimal network dependencies:

- 3x SSH relays, across 3 locations & 3 platforms
- Autossh: if the machine has power & Internet, we can get to it.
- SSH is blocked a lot less than VPNs in the countries where we work. 🤫
- Users are managed declaratively using JSON that is parsed using Nix.

Declarative access control helps prevent config drift



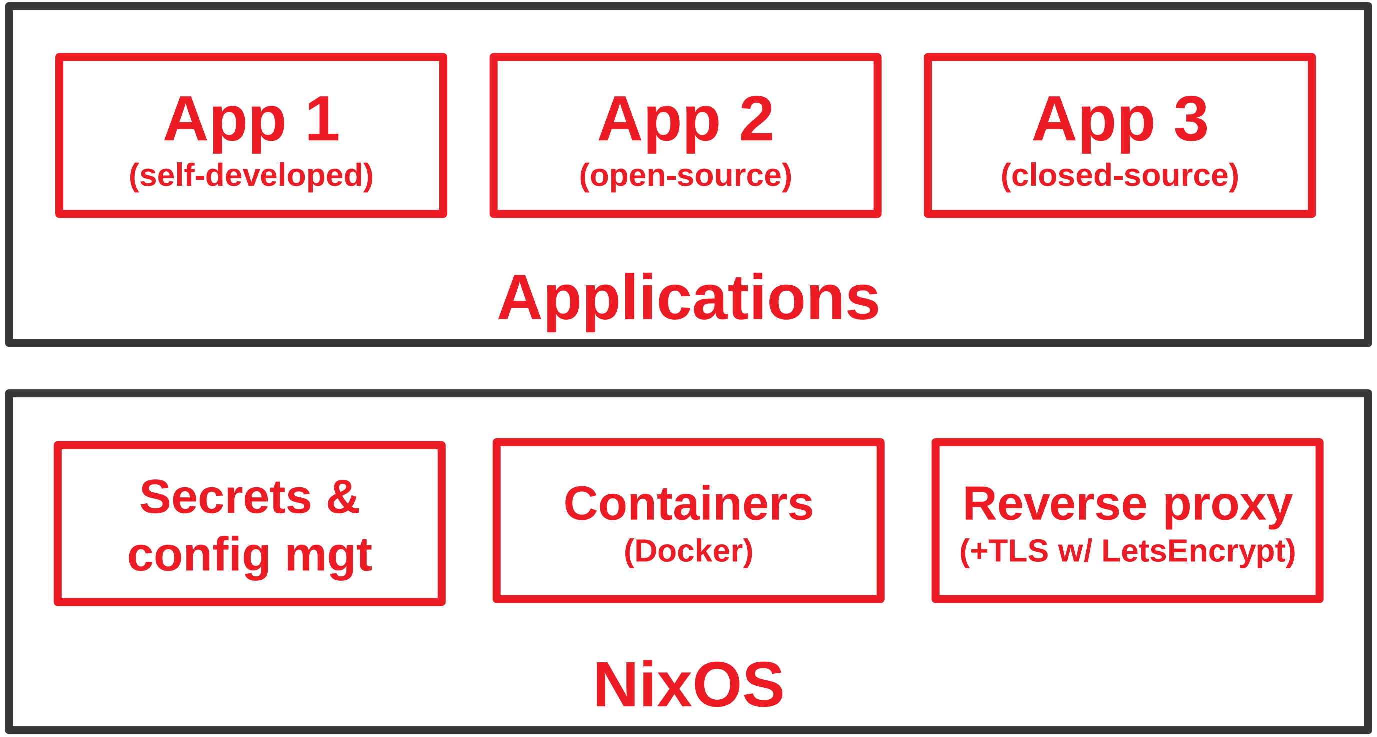
```
"docker-dmz-3": {
  "enable": {
    "nour":     "devops",
    "zbx_tnl": "remoteTunnel"
  },
  "enable_roles": [
    "devops",
    "hq",
    "infra_hq"
  ]
},
```



Containerized application deployments

<u>Developers, devops engineers and suppliers</u> provide a Docker image and config/secrets in YML

<u>The platform</u> is the same in on-prem, cloud and remote deployments.

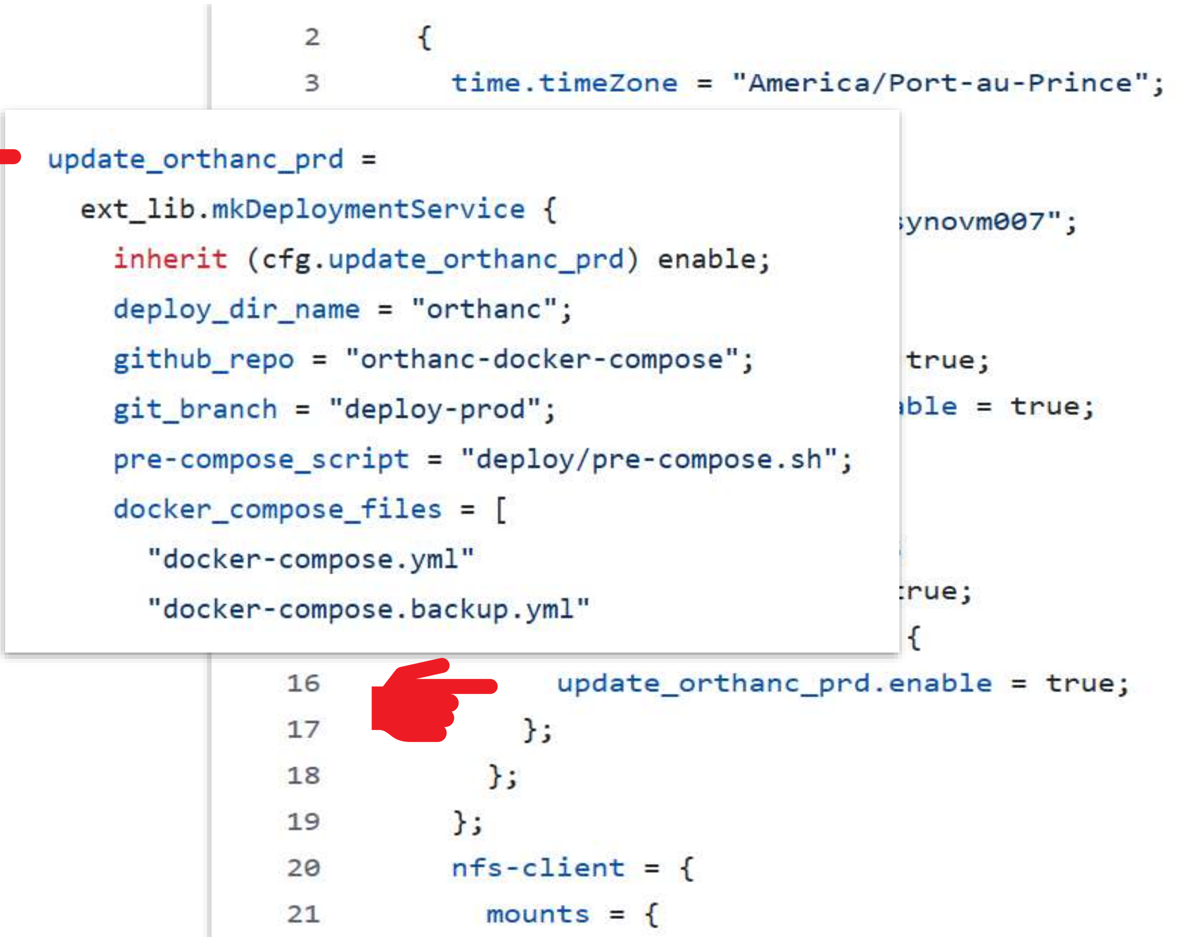


Automated application deployment

This service:

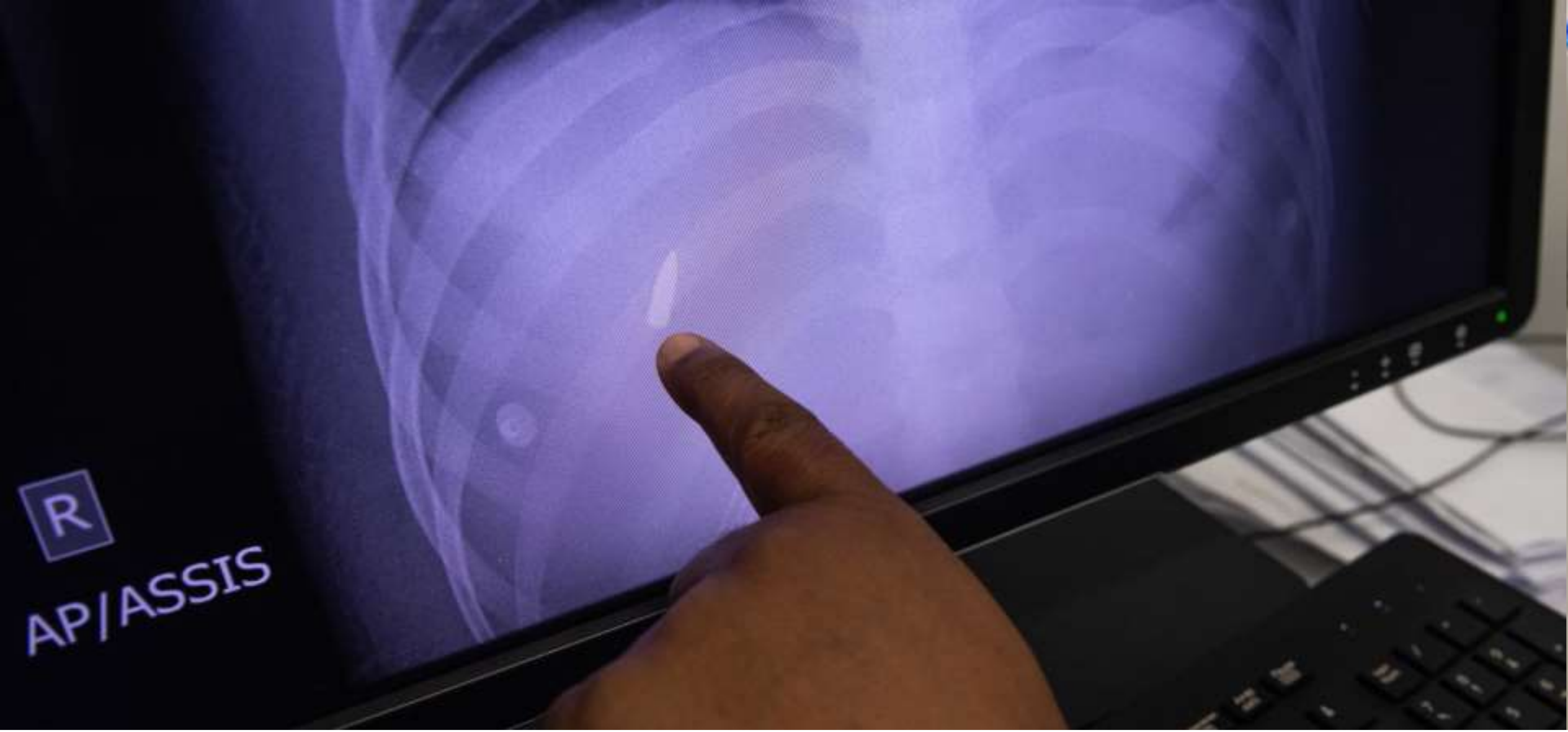
- Checks out a GitHub repo to a directory under /opt
- Optionally executes a shell script to e.g., pull fresh images or regenerate .env file
- Calls docker-compose up with a few arguments.

…That's basically it.



```
update_orthanc_prd =
  ext_lib.mkDeploymentService {
    inherit (cfg.update_orthanc_prd) enable;
    deploy_dir_name = "orthanc";
    github_repo = "orthanc-docker-compose";
    git_branch = "deploy-prod";
    pre-compose_script = "deploy/pre-compose.sh";
    docker_compose_files = [
      "docker-compose.yml"
      "docker-compose.backup.yml"
```

```
  {
    time.timeZone = "America/Port-au-Prince";
ynovm007";
true;
ble = true;
rue;
{
          update_orthanc_prd.enable = true;
        };
      };
    };
    nfs-client = {
      mounts = {
```



X-Ray showing a large calibre projectile lodged in a patient's rib cage. Stray bullets are a growing problem in Port-au-Prince. © Johnson Sabin

How do we provision servers?

A shell script:

- **Nixos-anywhere** installs the base system
- **Disko** for disk partitioning & formatting declaratively.
- Can optionally be enrolled in GitHub repo afterwards.
- **LUKS2** encrypted /opt & /home with a micro-app for emergency disk lock (invalidates the luks keyslot & reboots)

What could we improve?

- Sops-nix instead of ansible vault (sops-nix wasn’t available in 2018!)
- Better handling of encryption keys (SSH + secrets)
- Migrate to systemd-initrd ASAP
- Verified boot, use the TPM with measured boot for the encryption keys (or maybe remote attestation?)
- More VM tests
- Legacy code to refactor
  - Decouple modules
  - Less `with`
- Do everything on the build server then copy & switch without eval on-host.

MEDECINS SANS FRONTIERES

Issues we’ve faced

It’s not always plain sailing 🙂

- **Onboarding** new people to NixOS
- **Debugging** Nix code

Recommended:
NixOS in Production by Gabriela Gonzalez

Thank you very much for NixOS! It's an exceptional technology.

– Sohel & Ian

Special thanks to Ramses & Numtide!

Acknowledgements & feedback

If we could choose one thing to ask for 😁

~~Non-experimental flakes~~

Questions