Secrets are just encrypted config



Iulian Meghea

plumelo

STATUS QUO

Config management

Imperative → Declarative

ANSIBLE

config management

```
- name: install nginx
  apt: name=nginx state=present
```

NIXOS

config-driven operating system

```
services.nginx.enable = true;
```

TERRAFORM

infrastructure as code

```
resource "aws_s3_bucket" "prod" {
  bucket = "prod-storage"
}
```

STATUS QUO

Secrets management

Still imperative

FRAGILE

.env files

```
$ vi .env
```

UNAUDITED

password managers · memory

```
$ history | grep -i pass
```

DISCONNECTED

Vault / KeyVault

```
$ vault kv get -mount=kv prod/db
```

Cloudflare R2 went down. For an hour. Globally.



Production ran an old secret version. The backend had rotated to the new one.

Secret drift is worse. You find out from the outage.

“This incident happened because of human error and lasted longer than it should have because we didn’t have proper visibility into which credentials were being used by the R2 Gateway Worker to authenticate with our storage infrastructure.”

— Cloudflare R2 post-mortem · March 21, 2025

Cloudflare's own post-mortem

THE EQUATION

```
secrets = config + encryption
```

Same repo. Same commit. Same deploy. Same rollback.
The only thing added: encrypted at rest, decrypted at the edge.

How it works.

sops · age · sops-nix · systemd-credentials

.sops.yaml

```
keys:
  - &alice age1ql3z…
  - &helios age1f9x2…
  - &orion age1m4x7k…

creation_rules:
  - path_regex: servers/helios/[^/]+\.yaml$
    key_groups:
      - age: [*alice, *helios]
  - path_regex: servers/orion/[^/]+\.yaml$
    key_groups:
      - age: [*alice, *orion]
  - path_regex: shared/[^/]+\.yaml$
    key_groups:
      - age: [*alice, *helios, *orion]
```

key.txt

```
alice
# public key: age1ql3z7ys6hh5kxd4xs3x2c0syye5efv…
AGE-SECRET-KEY-1W8DQQQQQQQQQQQQQQ…

helios
# public key: age1f9x2kz7qf9x2c0syye5efv3q4…
AGE-SECRET-KEY-1QYPQQQQQQQQQQQQQQ…

orion
# public key: age1m4x7k9tq6hh5kxd4xs3x2c0syy…
AGE-SECRET-KEY-1KZ7QQQQQQQQQQQQQQ…
```

orion can't read helios secrets — the cryptography enforces it.

servers/helios/secrets.yaml — plain

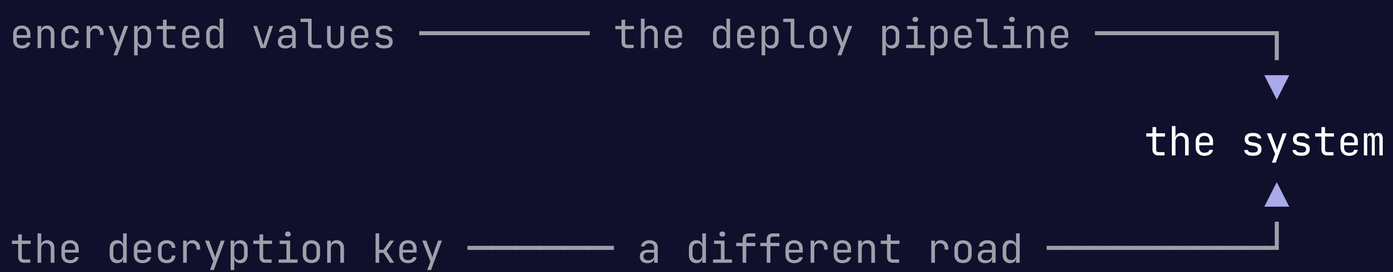
```
data:
  db_password: Jh7%ne2Lpq
  smtp_password: Kx2!mq8Zv
  api_token: yn4-wQ93mB
```

servers/helios/secrets.yaml — after sops

```
data:
  db_password: ENC[AES256_GCM,data:9tQLx…]
  smtp_password: ENC[AES256_GCM,data:vHm2p…]
  api_token: ENC[AES256_GCM,data:kQ8wN…]

sops:
  age:
    - recipient: age1ql3z…   # the dev key
    - recipient: age1f9x2…   # helios
```

Still diffs like config.

also: json · env · ini · binary



Deterministic. Declarative. Reproducible.

incus · qemu · air-gapped

pull request

```
[bot] rotate db password — helios
status: open · reviewer: alice — approved
servers/helios/secrets.yaml

- db_password: ENC[AES256_GCM,data:9tQLx…]
+ db_password: ENC[AES256_GCM,data:kR4wN…]
```

git log

```
$ git log --pretty=format:"%h | %an | %ar | %s"
  servers/helios/secrets.yaml

a1b2c3 | alice | 12 min ago   | rotate db password
d4e5f6 | bob   | 3 months ago | rotate db password
g7h8i9 | alice | forever ago  | initial import
```

You know **when**. You know **who**. You know **which systems** have it.

The audit trail is git history. Nothing extra to dig through. No API calls to trace.

HOW IT WORKS: FULL NIXOS EXAMPLE

key delivery

```
# terraform — key delivery
provisioner "local-exec" {
  command = <<-EOT  (1)
  sops -d --extract '["age"]["key"]' secrets.yaml |
  ssh root@helios 'cat > /var/lib/sops-nix/key.txt'
  EOT
}

# nixos
sops = {
  defaultSopsFile = ./secrets.yaml;
  age.keyFile = "/var/lib/sops-nix/key.txt";  (2)
};
sops.secrets."db_password" = { };
```

the service

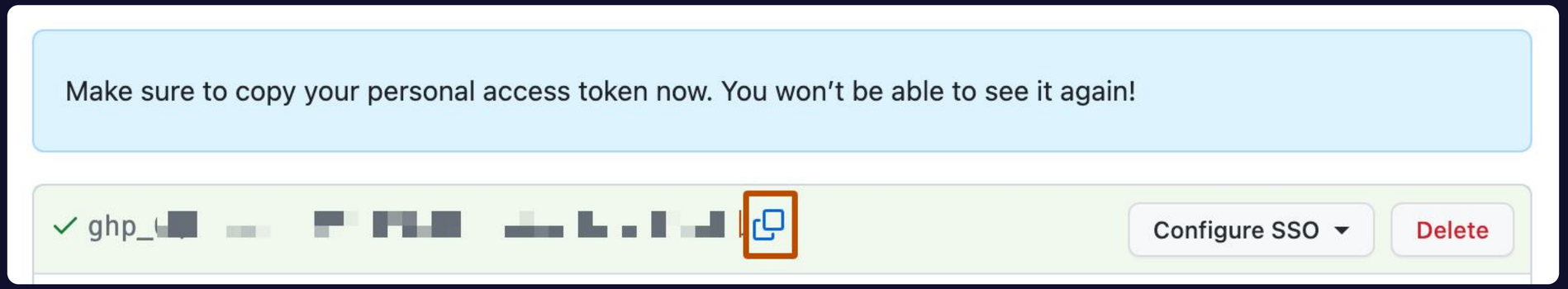
```
sops.templates."app/env" = {
  content = ''  (3)
  DB_PASSWORD=${config.sops.placeholder."db_password"}
  '';
};

systemd.services.app = {
  serviceConfig.DynamicUser = true;  (4)
  serviceConfig.LoadCredential =  (5)
    "env:${config.sops.templates."app/env".path}";
  script = ''
    app --env-file \
      $CREDENTIALS_DIRECTORY/env  (6)
  '';
};
```

Provision the key. Declare the secret. Isolate the service.

terraform · sops-nix · systemd-credentials

Still, some friction remains.

click-ops · passwords



- can't generate it ahead of time
- can't use a better algorithm
- can't adjust length
- can't rotate it

You can only record it.

FRICTION: PASSWORD ZERO

the machine

git + sops
db creds · api tokens · webhooks
machine identity keys
decrypted at activation
no human in the loop

the human

your age key
plaintext by design — no passphrase
lives inside the vault
or on hardware — touch to unlock
one master password, in your head

Is the age key a secret or a password?

secrets ← age key ← vault ← your head

Same outage.
Different ending.

```
# the cloudflare drift, rewritten
$ git revert HEAD
```

one commit — one rollback. app and secret, together.

SYSTEMS

CONFIG

declarative
terraform + nixos

SECRETS

declarative
sops + age

DATA

restic + /var/lib

config, secrets & data

SYSTEMS

CONFIG

declarative - reproducible · in git

DATA

restic + /var/lib

Just config and data.

FUTURE

TPM

keys sealed to the machine

IN MEMORY

never at rest

TRANSPORT

how keys reach the servers

What's next?

Agents can’t leak what they never had.

WHAT AGENTS GET TODAY

agent shell

```
$ env | grep -iE 'key|token|pass'
DB_PASSWORD=Jh7%ne2Lpq
SMTP_PASSWORD=Kx2Iqg8Zv
API_TOKEN=yn4-wQ93mB
AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
```

WHAT THEY SHOULD SEE

agent view

```
$ cat secrets.yaml
db_password:   ENC[AES256_GCM,…]
smtp_password: ENC[AES256_GCM,…]
api_token:     ENC[AES256_GCM,…]
```

The secrets are encrypted values in git — the agent sees the shape, never the value.

THANK YOU

Secrets are just encrypted config

www.linkedin.com/in/iulian-meghea



Iulian Meghea

plumelo