

Managing server secrets at scale

with a vaultless password manager

Ignat Korchagin

$ whoami

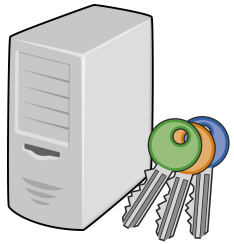
- Platform engineer at Cloudflare
- Passionate about security and crypto
- Enjoy low level programming



Disclaimer

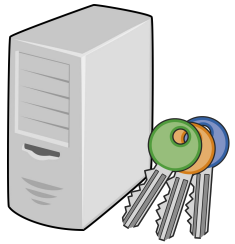


So you have a server



CLOUDFLARE

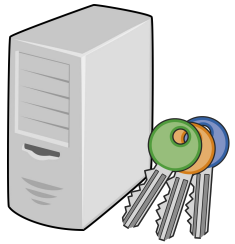
So you have a server



You need:

- server SSH key

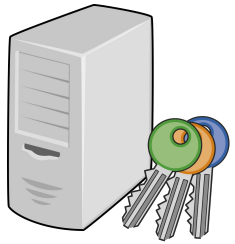
So you have a server



You need:

- server SSH key
- configuration management key

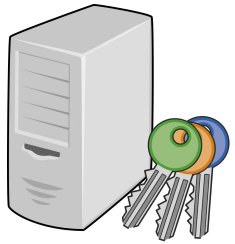
So you have a server



You need:

- server SSH key
- configuration management key
- disk encryption key

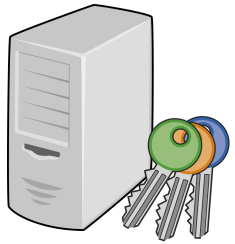
So you have a server



You need:

- server SSH key
- configuration management key
- disk encryption key
- some server credentials

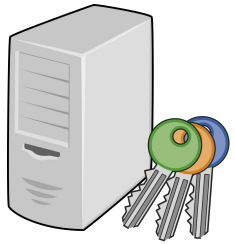
So you have a server



You need:

- server SSH key
- configuration management key
- disk encryption key
- some server credentials
- probably more...

So you have a server



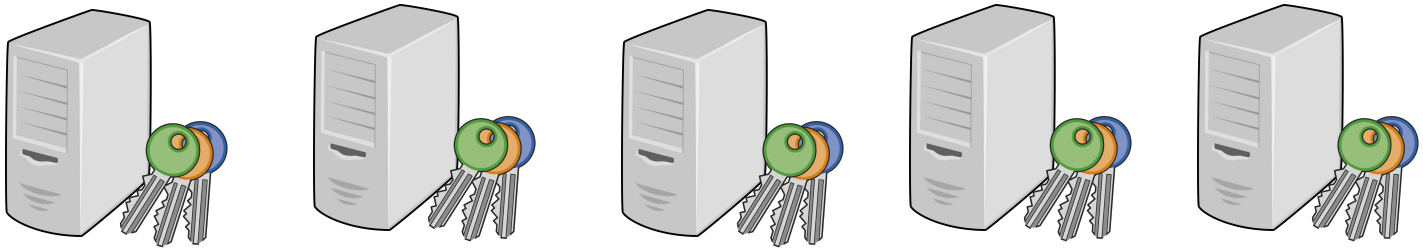
You need:

- server SSH key
- configuration management key
- disk encryption key
- some server credentials
- probably more…

… so at least 5 keys… and that’s per server

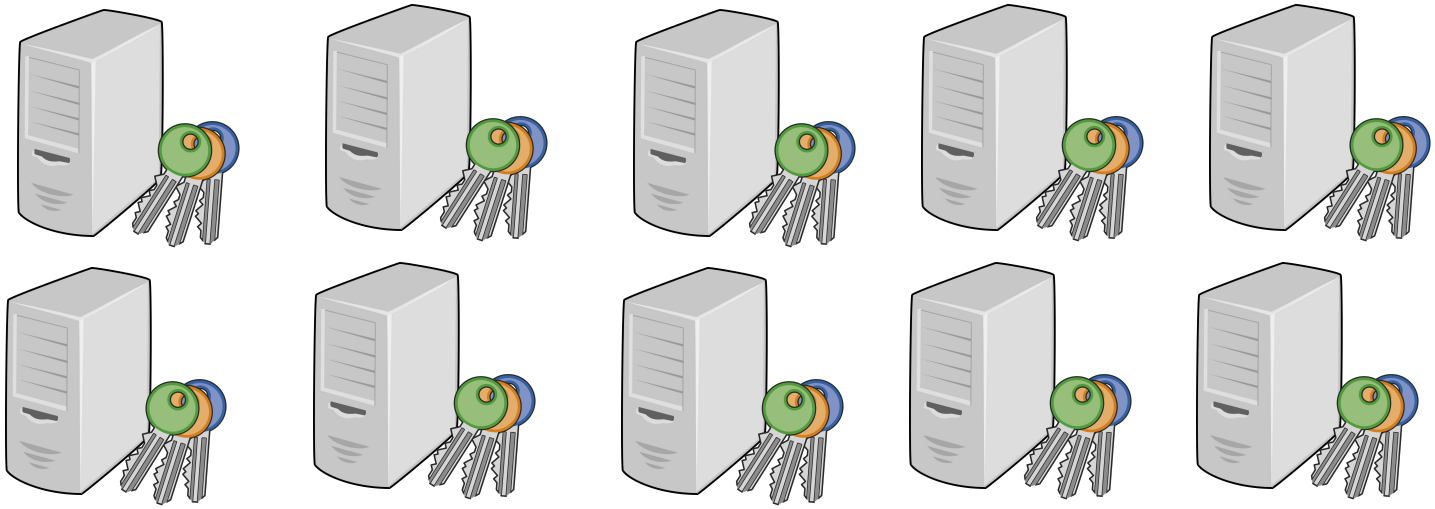
CLOUDFLARE

So you have a datacentre



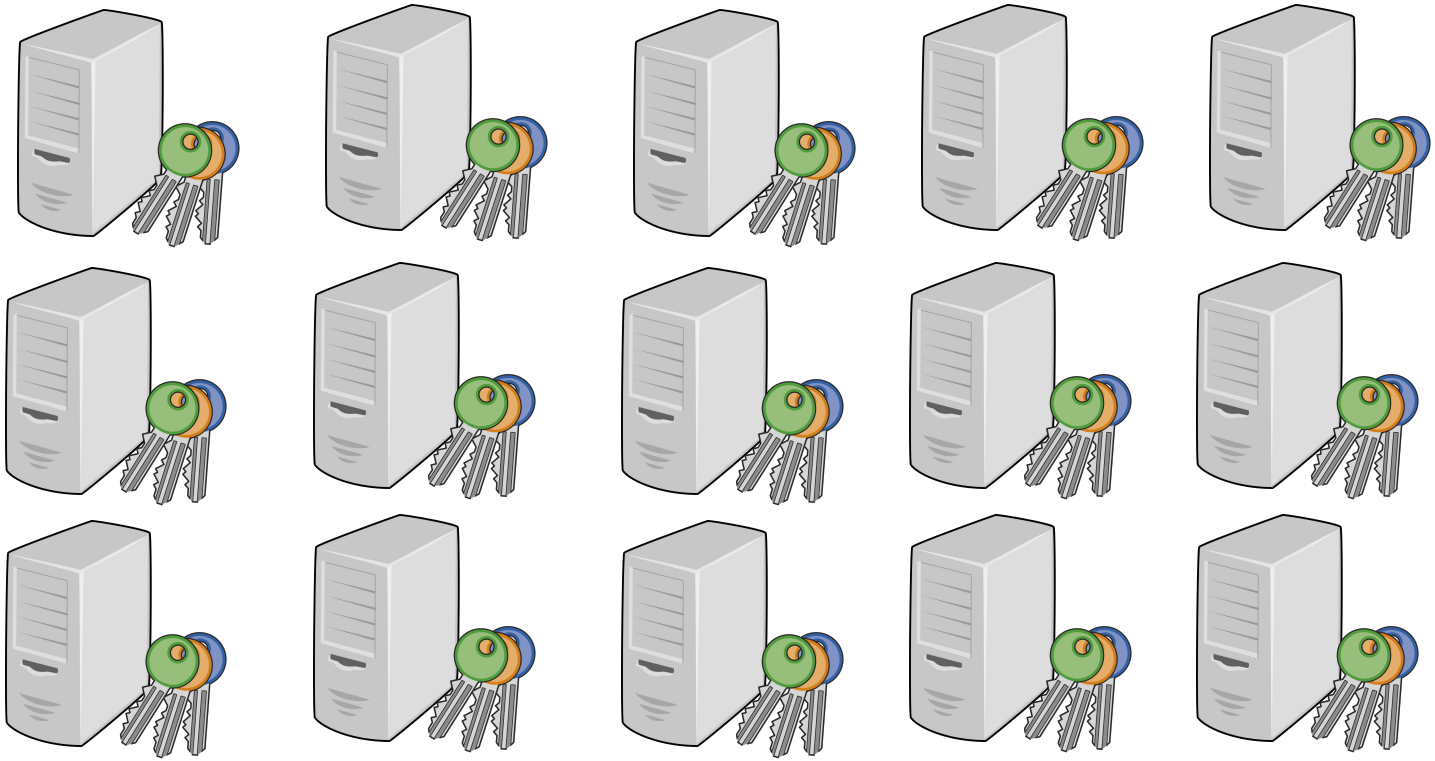
CLOUDFLARE

So you have a datacentre**(s)**



CLOUDFLARE

So you have a datacentre**(s)**



CLOUDFLARE

Cloudflare network today



CLOUDFLARE

So you have a datacentre**(s)**



CLOUDFLARE

Where do keys live?

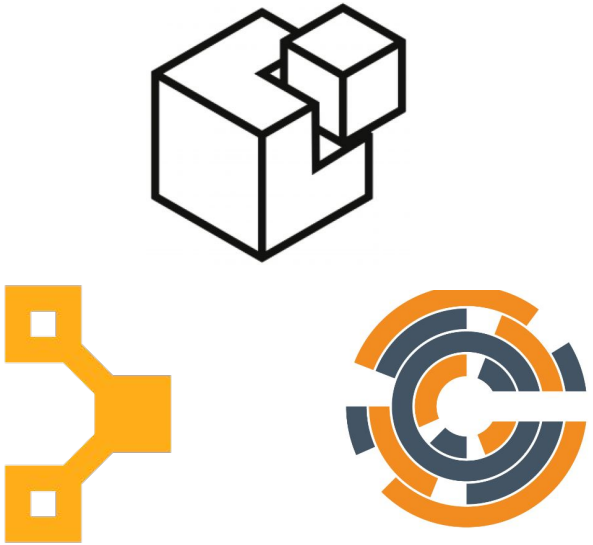


Where do keys live?



Where do keys live?





Keys in configuration management

```
#!yaml|gpg

root-password: |
    -----BEGIN PGP MESSAGE-----
    Version: GnuPG v1

    hQIMA221uplZYlMdARAA583Z4o3xZawWzK8yUYJKBEkMQD/i+RRn7A0+h8SEmsov
    QkrxgeaCWfIZ5pRpCpVOK1SWZGi0dzWkWe1DeNisawv5X/VUG3d5ej1xtAD4kBTy
    AzcnFft7QfIsV8C+jguHYGITU++pFVAgEdGrb09mf6SEDaAGJhOq01BmHccw0Pat
    rBH/+gvD155F7sxM/BBQwL25ZjtC+8jUsp1bUcTQVofsy6kTVRNSS4hO4UNtMuMQ
    hYf6UAPaJv3PhFXKYYu0tEp2THZVTlUtTjyKAZrNiKyRpC/0exbJjJMqkYmmUG9r
    yP1CvubJnmHda2u42981dK3pz5TlLEO4MrBry6vynN0TJfXwn1nt7YMVativiQb9
    UK5NDbjVKBBE6KkN28kSJtsTkCOM7+RztjLdf+7ZWzwxFV5EkM+2SLPIhQFCMjRG
    ...
```

CLOUDFLARE

Keys in configuration management



CLOUDFLARE

Keys in configuration management

- Bootstrap configuration management



CLOUDFLARE

Keys in configuration management

- Bootstrap configuration management
- Does not scale for unique keys



Keys in configuration management

- Bootstrap configuration management
- Does not scale for unique keys



```
#!yaml|gpg

root-password: |
{% if grains['hostname'] == 'baredog' %}
  -----BEGIN PGP MESSAGE-----
  ...
{% elif grains['hostname'] == 'cheesyonion' %}
  -----BEGIN PGP MESSAGE-----
  ...
{% elif grains['hostname'] == … %}
```

CLOUDFLARE

Keys on local disk

Keys on local disk

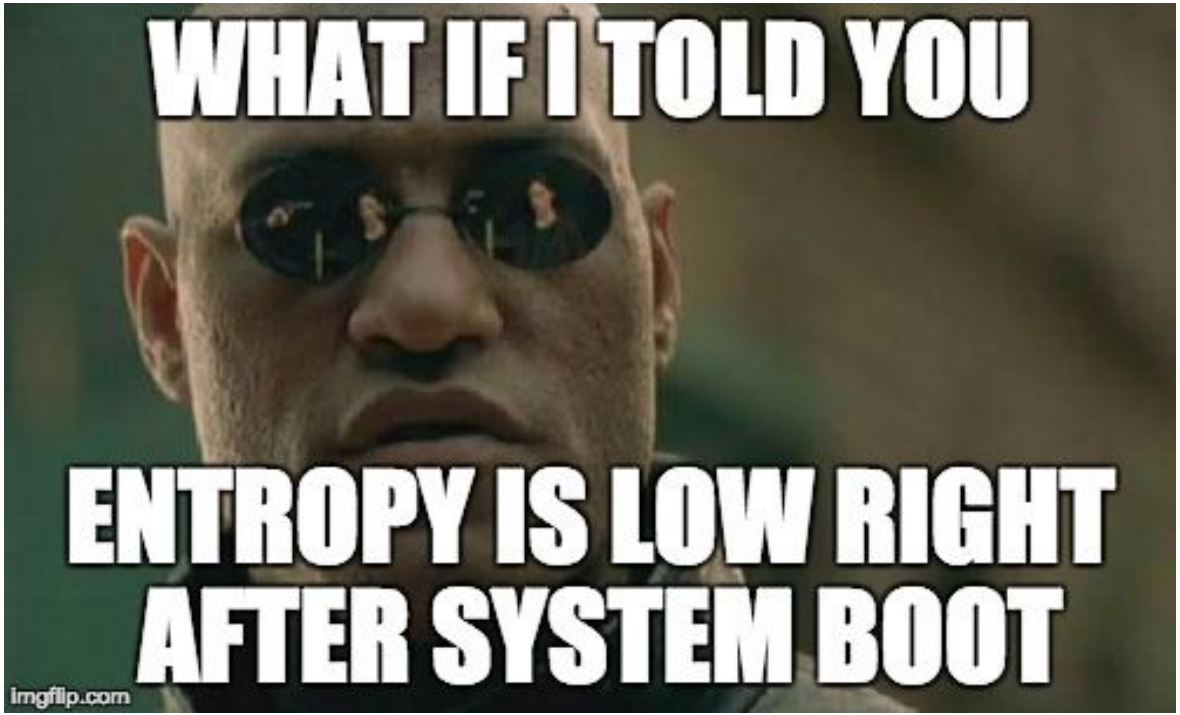
- Usually generated by startup scripts

```
#!/bin/bash

# super nitty startup script, fully automated !!!
if [ ! -f /etc/server_key ]; then
  dd if=/dev/urandom of=/etc/server_key bs=1 count=32
fi

# don't forget SSH
if [ ! -f /etc/ssh/id_rsa ]; then
  ssh-keygen -f /etc/ssh/id_rsa
fi
```

Entropy



CLOUDFLARE

Keys on local disk



CLOUDFLARE

Keys on local disk

- Not suitable for some key types



CLOUDFLARE

Keys on local disk

- Not suitable for some key types
  - root passwords



CLOUDFLARE

Keys on local disk

- Not suitable for some key types
  - root passwords
- Does not play well with disk encryption



Keys on local disk

- Not suitable for some key types
  - root passwords
- Does not play well with disk encryption
  - decrypt configuration management key



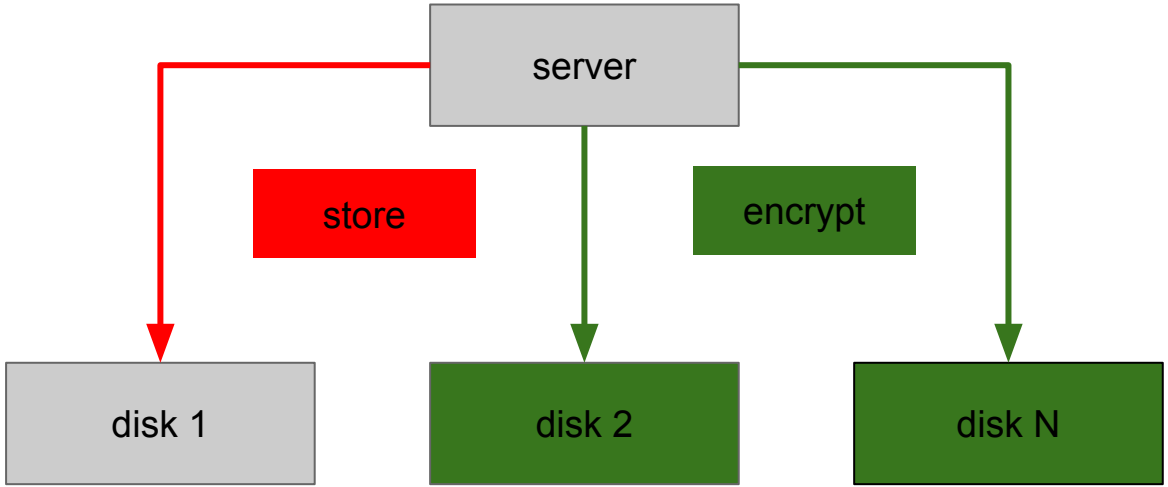
Keys on local disk

- Not suitable for some key types
  - root passwords
- Does not play well with disk encryption
  - decrypt configuration management key
- What about diskless/stateless systems?



CLOUDFLARE

Encrypted disks



Unified Extensible Firmware Interface



CLOUDFLARE

Unified Extensible Firmware Interface

- Aka BIOS 2.0

Unified Extensible Firmware Interface

- Aka BIOS 2.0
- Standard pre-OS environment

Unified Extensible Firmware Interface

- Aka BIOS 2.0
- Standard pre-OS environment
- Extensible (you can write your own apps)

Unified Extensible Firmware Interface

- Aka BIOS 2.0
- Standard pre-OS environment
- Extensible (you can write your own apps)
- Supported by most major OSes

Unified Extensible Firmware Interface

- Aka BIOS 2.0
- Standard pre-OS environment
- Extensible (you can write your own apps)
- Supported by most major OSes
- Provides many advanced features

Unified Extensible Firmware Interface

- Aka BIOS 2.0
- Standard pre-OS environment
- Extensible (you can write your own apps)
- Supported by most major OSes
- Provides many advanced features
  - UEFI variables

UEFI variables

- Backed by flash memory on platform firmware chip

UEFI variables

- Backed by flash memory on platform firmware chip
- Can store standard and custom (OEM/user) data

UEFI variables

- Backed by flash memory on platform firmware chip
- Can store standard and custom (OEM/user) data
- Can be accessed after OS kernel booted

UEFI variables

- Backed by flash memory on platform firmware chip
- Can store standard and custom (OEM/user) data
- Can be accessed after OS kernel booted
- Have built-in support in Linux

UEFI variables in Linux

```
# not needed for systemd-based Linux distributions
mount -t efivarfs efivarfs /sys/firmware/efi/efivars


# need to prepend data with 4 byte attr and put an “owner” GUID
cat <(printf "\x07\x00\x00\x00") <(cat mydata.bin) > \
/sys/firmware/efi/efivars/mydata-<some GUID>
```

CLOUDFLARE

UEFI variables in Linux

```
# not needed for systemd-based Linux distributions
mount -t efivarfs efivarfs /sys/firmware/efi/efivars

# need to prepend data with 4 byte attr and put an “owner” GUID
cat <(printf "\x07\x00\x00\x00") <(cat mydata.bin) > \
/sys/firmware/efi/efivars/mydata-<some GUID>
```

- always available

UEFI variables in Linux

```
# not needed for systemd-based Linux distributions
mount -t efivarfs efivarfs /sys/firmware/efi/efivars

# need to prepend data with 4 byte attr and put an “owner” GUID
cat <(printf "\x07\x00\x00\x00") <(cat mydata.bin) > \
/sys/firmware/efi/efivars/mydata-<some GUID>
```

- always available
- can be accessed in early boot stages

CLOUDFLARE

UEFI variables in Linux

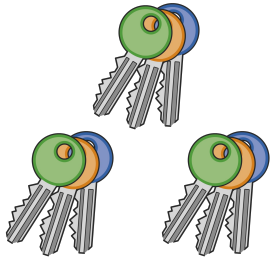
```
# not needed for systemd-based Linux distributions
mount -t efivarfs efivarfs /sys/firmware/efi/efivars

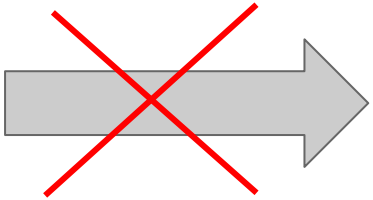

# need to prepend data with 4 byte attr and put an “owner” GUID
cat <(printf "\x07\x00\x00\x00") <(cat mydata.bin) > \
/sys/firmware/efi/efivars/mydata-<some GUID>
```

- always available
- can be accessed in early boot stages
- however, may have limited storage

CLOUDFLARE

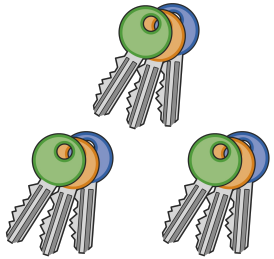
Keys in cryptography

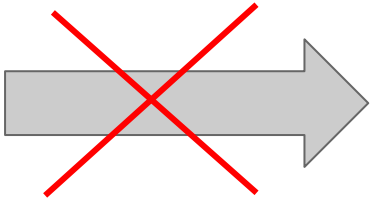


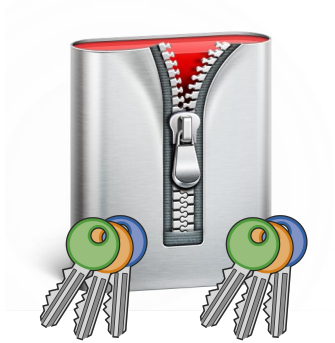


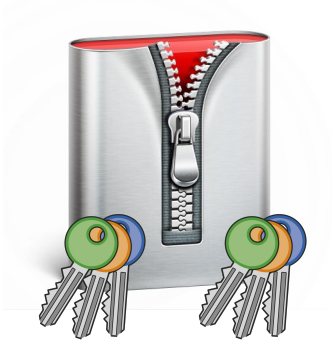


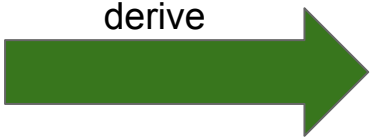
Keys in cryptography

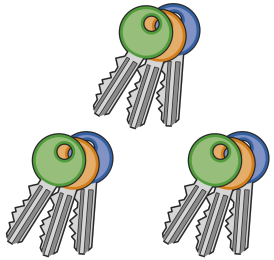












CLOUDFLARE

Keys in cryptography

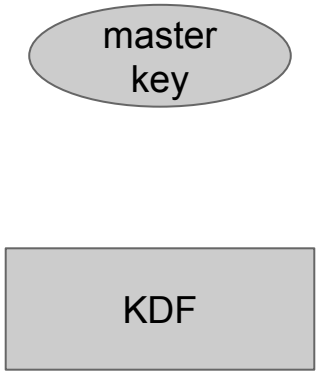




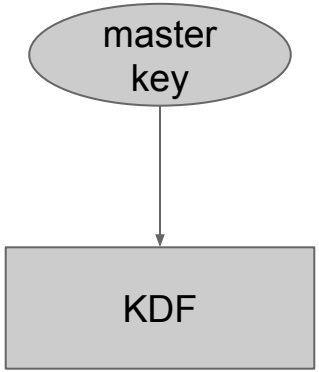


CLOUDFLARE

Key derivation functions

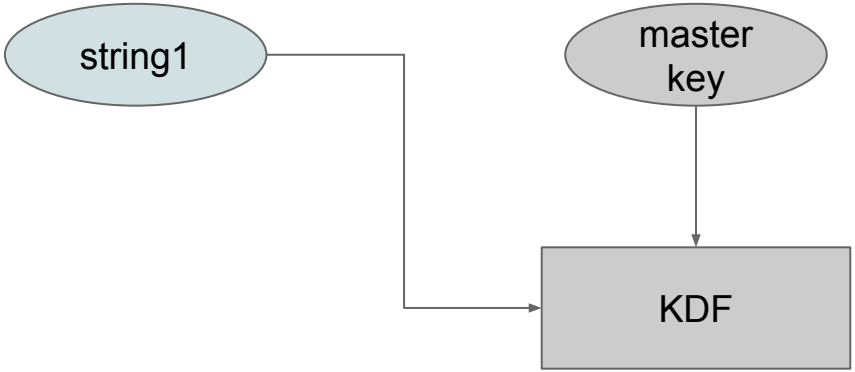


Key derivation functions

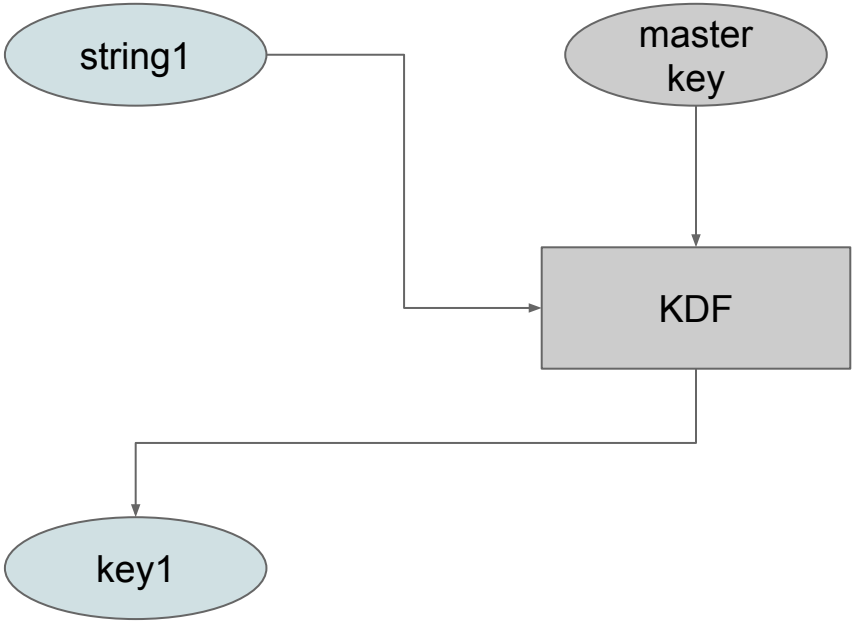


CLOUDFLARE

Key derivation functions

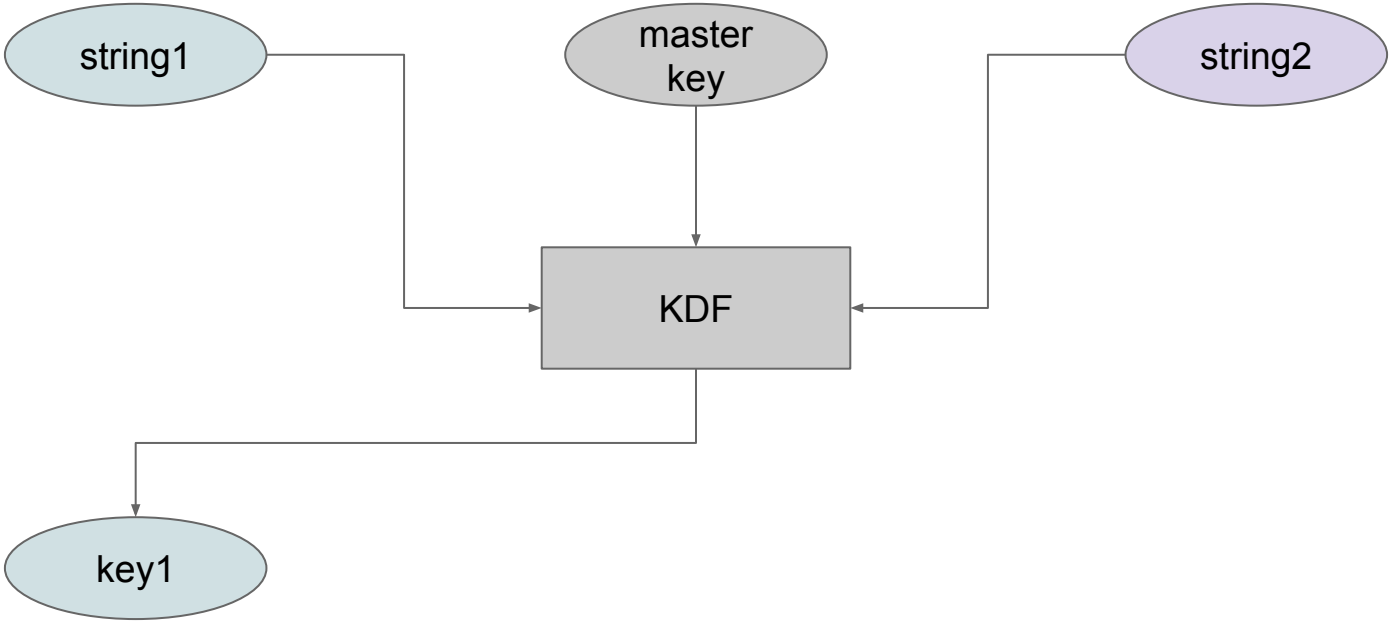


Key derivation functions



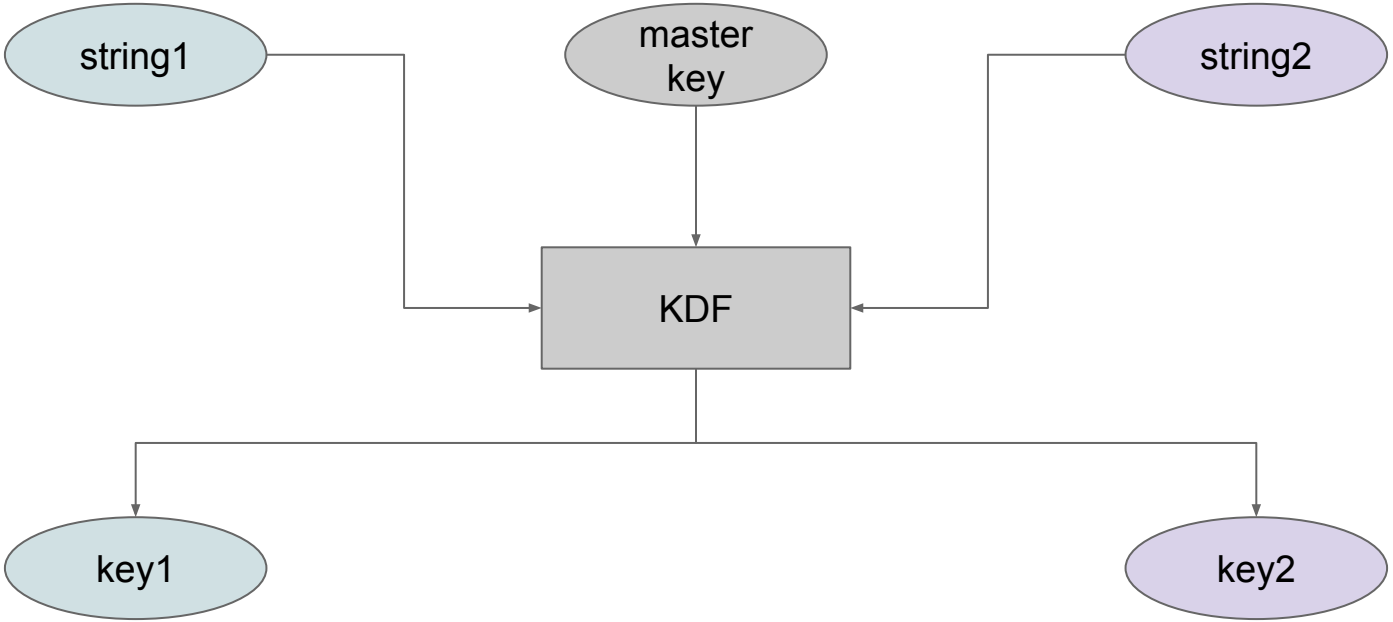
CLOUDFLARE

Key derivation functions



CLOUDFLARE

Key derivation functions



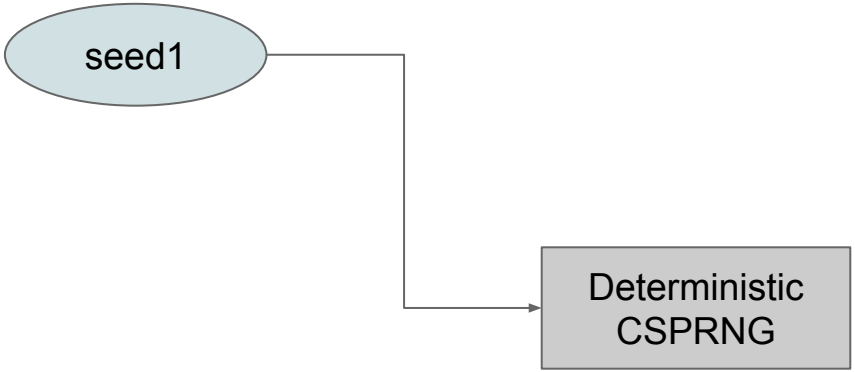
CLOUDFLARE

Generating key pairs

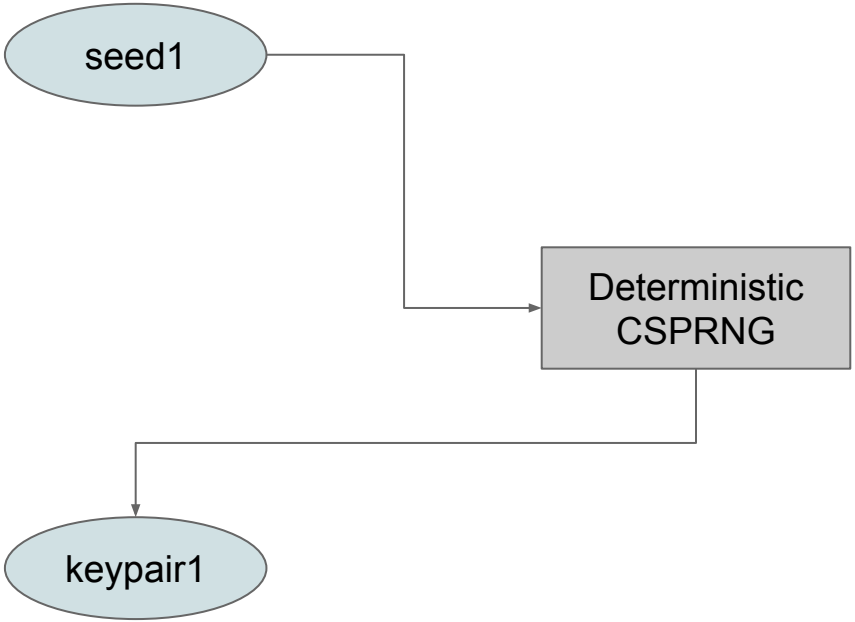


CLOUDFLARE

Generating key pairs

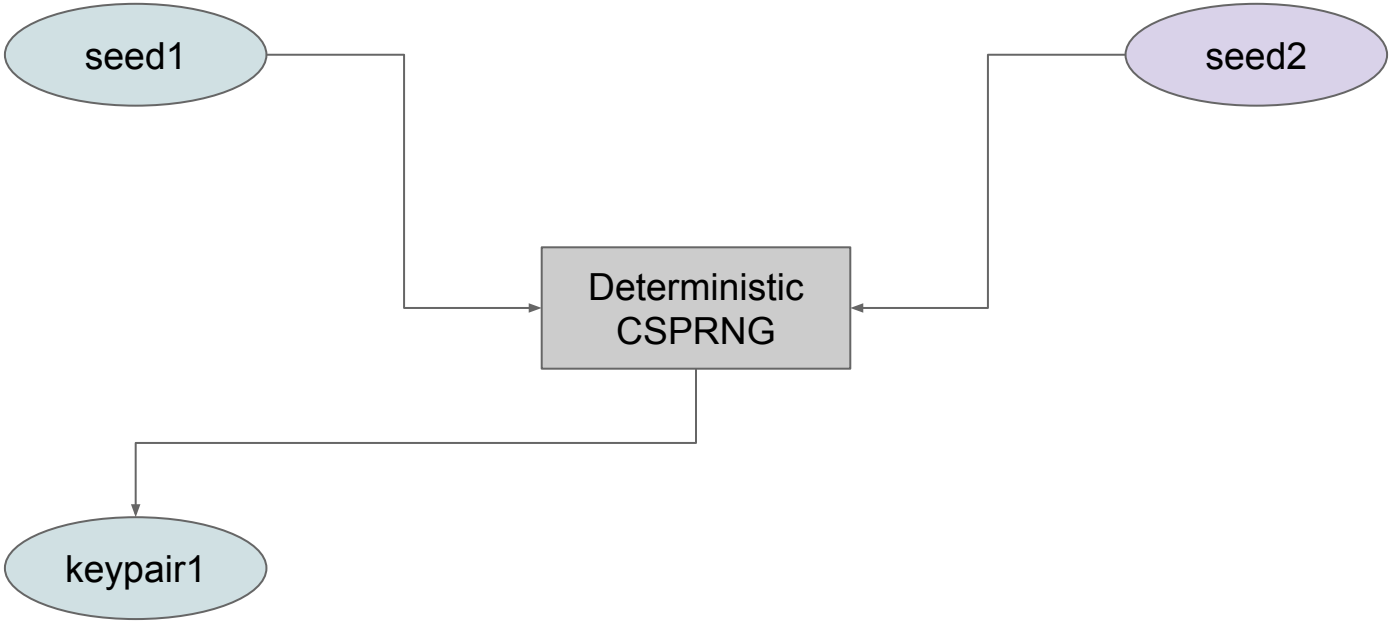


Generating key pairs



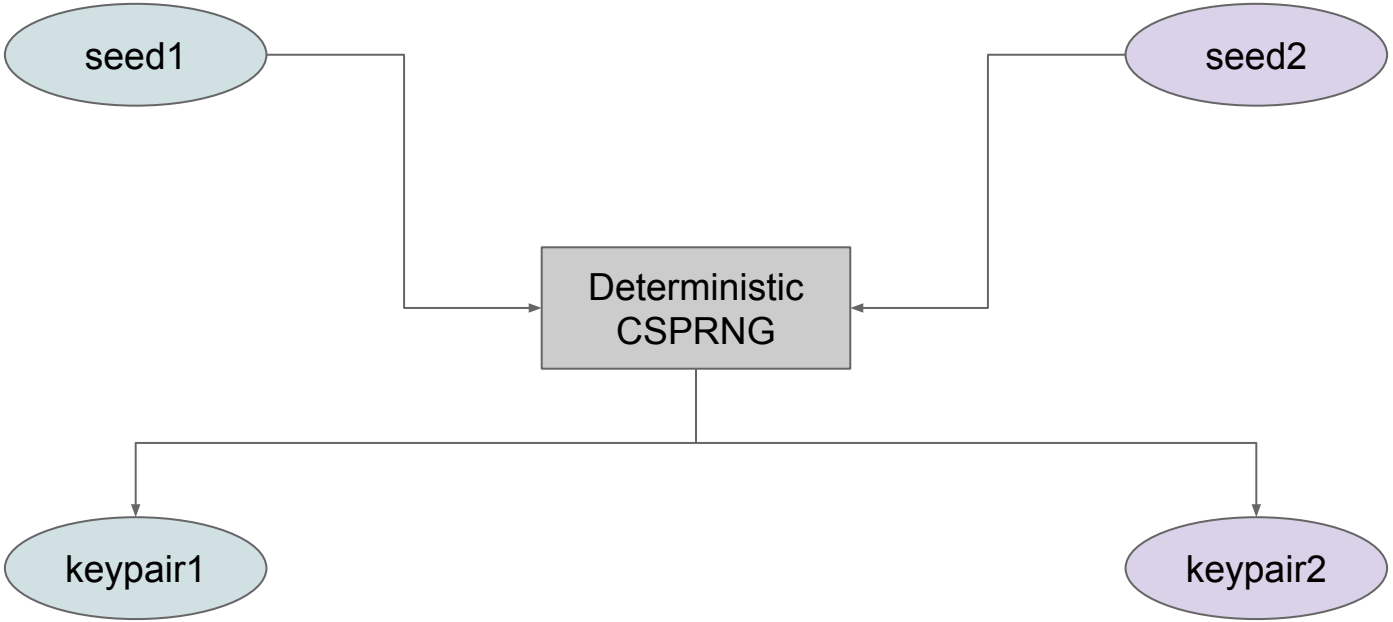
CLOUDFLARE

Generating key pairs



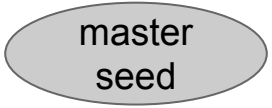
CLOUDFLARE

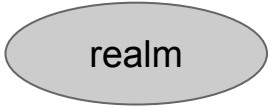
Generating key pairs



CLOUDFLARE

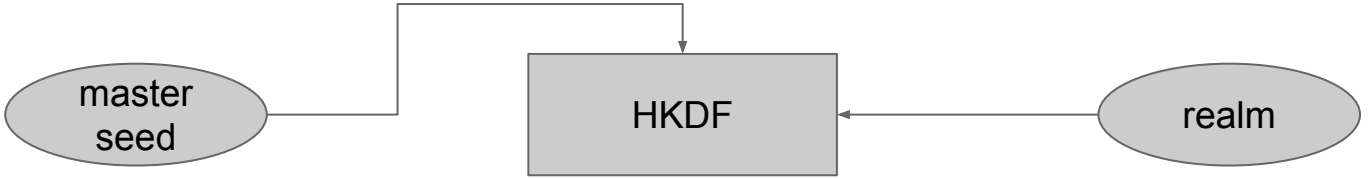
Introducing gokey tool





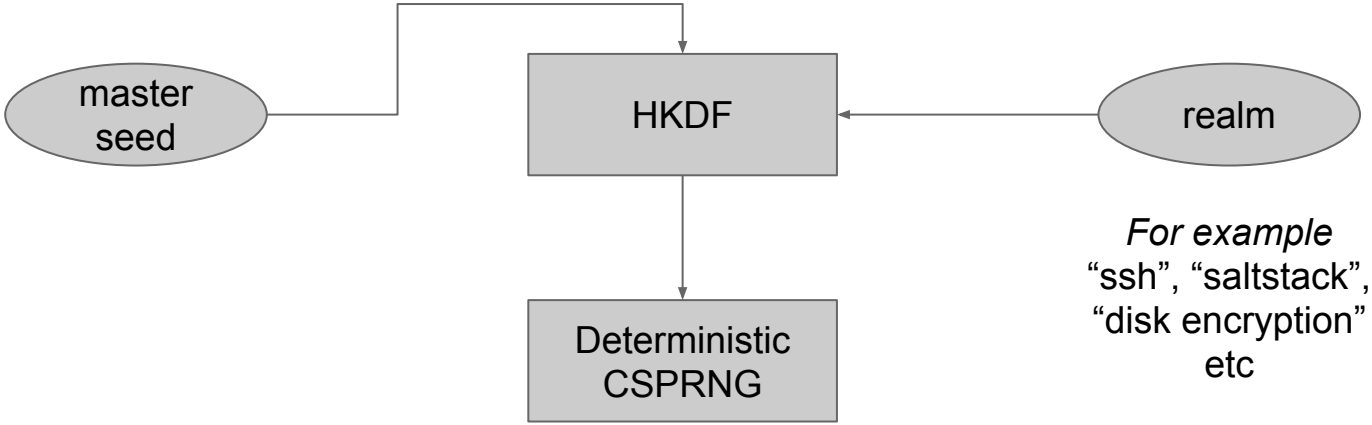
For example
“ssh”, “saltstack”,
“disk encryption”
etc

Introducing gokey tool



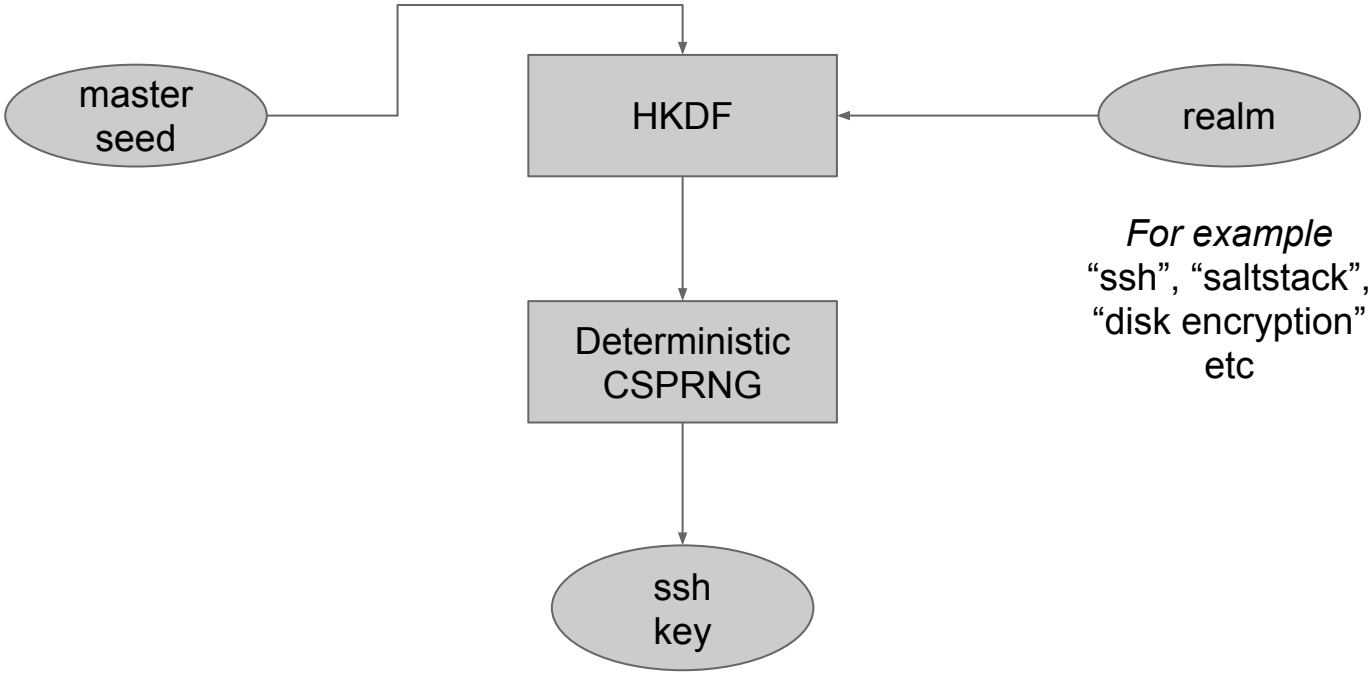
For example
“ssh”, “saltstack”,
“disk encryption”
etc

Introducing gokey tool

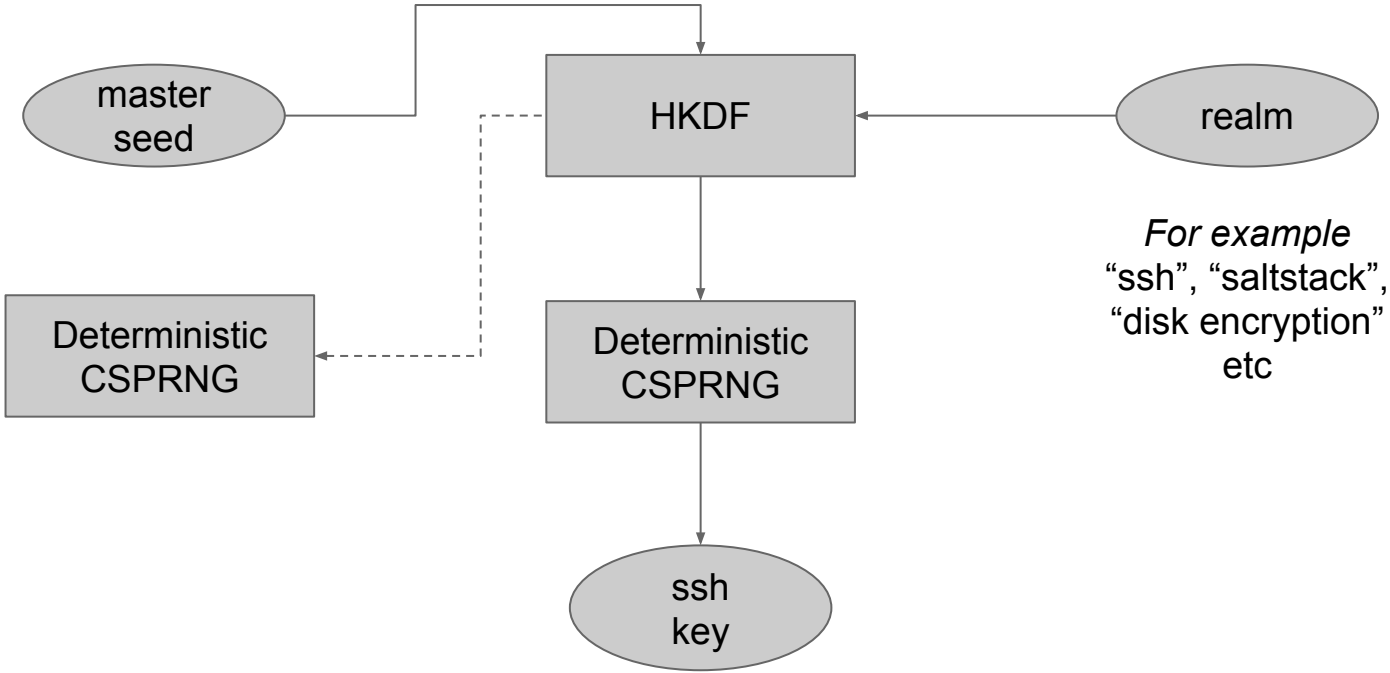


CLOUDFLARE

Introducing gokey tool

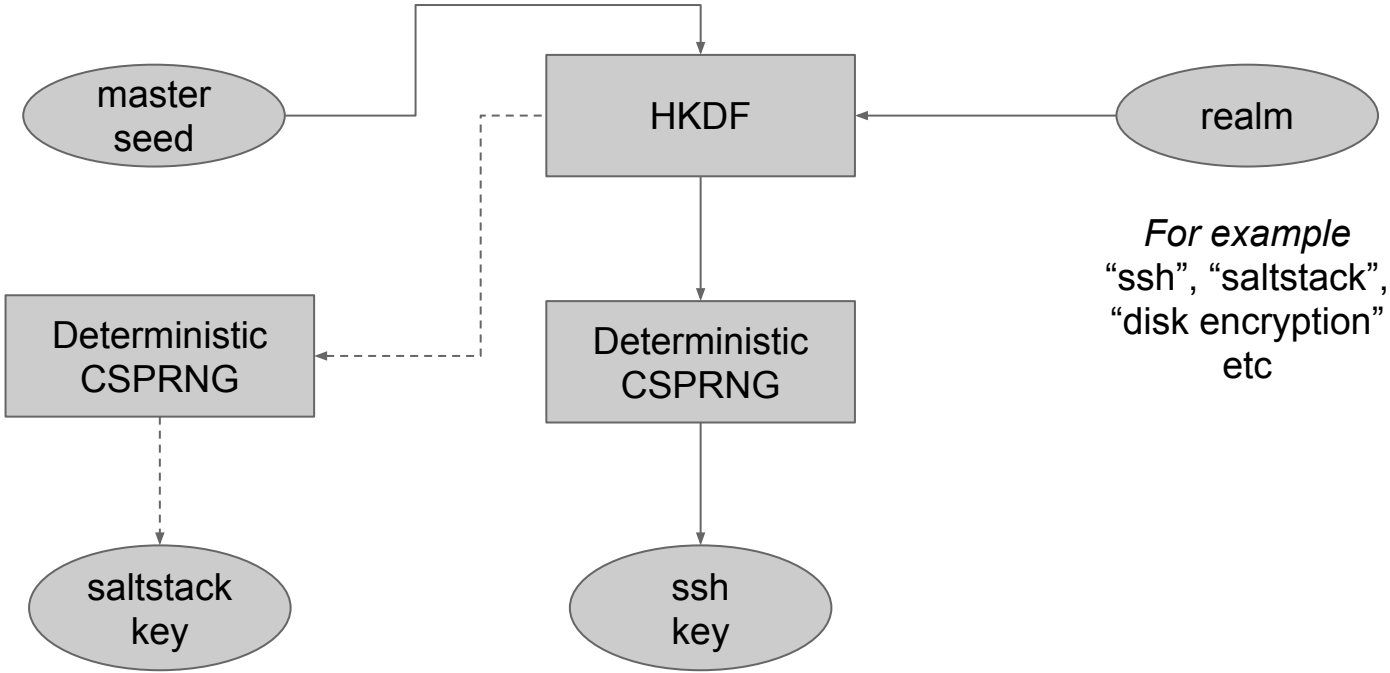


Introducing gokey tool



CLOUDFLARE

Introducing gokey tool



CLOUDFLARE

Key management

- Provisioning process ensures a master seed is generated and stored in UEFI on first boot

Key management

- Provisioning process ensures a master seed is generated and stored in UEFI on first boot
- Startup scripts “recover” (derive from master seed) configuration management credential (key)

Key management

- Provisioning process ensures a master seed is generated and stored in UEFI on first boot
- Startup scripts “recover” (derive from master seed) configuration management credential (key)
- Configuration management “recovers” all other keys

Key management

```
root-password: |
{% if grains['hostname'] == 'baredog' %}
  -----BEGIN PGP MESSAGE-----
  ...
{% elif grains['hostname'] == 'cheesyonion' %}
  -----BEGIN PGP MESSAGE-----
```

CLOUDFLARE

Key management

```
root-password: |
{% if grains['hostname'] == 'baredog' %}
  -----BEGIN PGP MESSAGE-----
  ...
{% elif grains['hostname'] == 'cheesyonion' %}
  -----BEGIN PGP MESSAGE-----
```



```
root-password: {% gokey('root-password') %}
ssh-key: {% gokey('ssh') %}
```

CLOUDFLARE

Key management

```
root-password: |
{% if grains['hostname'] == 'baredog' %}
  -----BEGIN PGP MESSAGE-----
  ...
{% elif grains['hostname'] == 'cheesyonion' %}
  -----BEGIN PGP MESSAGE-----
```

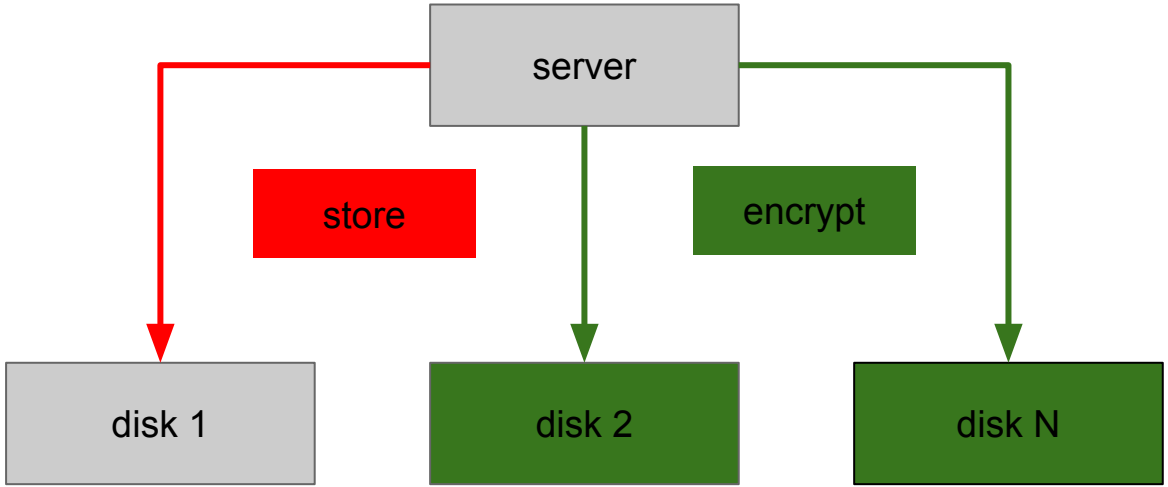


```
root-password: {% gokey('root-password') %}
ssh-key: {% gokey('ssh') %}
```

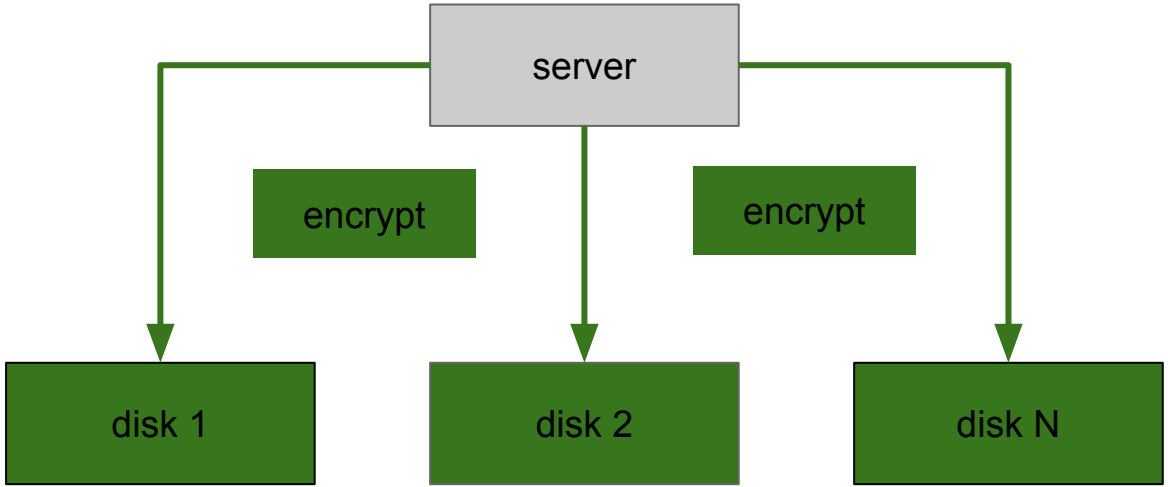
```
ssh-key: {% gokey('ssh-v2') %}
```

CLOUDFLARE

Encrypted disks (previously)



Encrypted disks



Conclusions

Conclusions

- Decouple key storage from regular storage

Conclusions

- Decouple key storage from regular storage
- Decouple key contents from key management

Conclusions

- Decouple key storage from regular storage
- Decouple key contents from key management
- Easy to add new or rotate existing keys

Conclusions

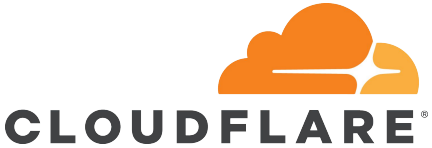
- Decouple key storage from regular storage
- Decouple key contents from key management
- Easy to add new or rotate existing keys
- Better security guarantees

Conclusions

- Decouple key storage from regular storage
- Decouple key contents from key management
- Easy to add new or rotate existing keys
- Better security guarantees

https://github.com/cloudflare/gokey

CLOUDFLARE



Thank you