Secrets

Manage, Deploy and Store

Sven Sternberger

Hepix 2014 Annecy

HELMHOLTZ
ASSOCIATION

DESY

Examples of secrets

- Passwords
- Certificates
- Private Keys
- Kerberos Keytab

Manage

- Ensure Quality of password
- Control lifetime
    - certificate
    - password
- Control secret distribution
- Control secret access

Deploy

- Distribute secrets to selected hosts
- Update secrets on selected hosts
- Control secrets on selected hosts

Store

- Central Place
- Controlled Environment
- Accessible to admins
- Delegate rights to groups

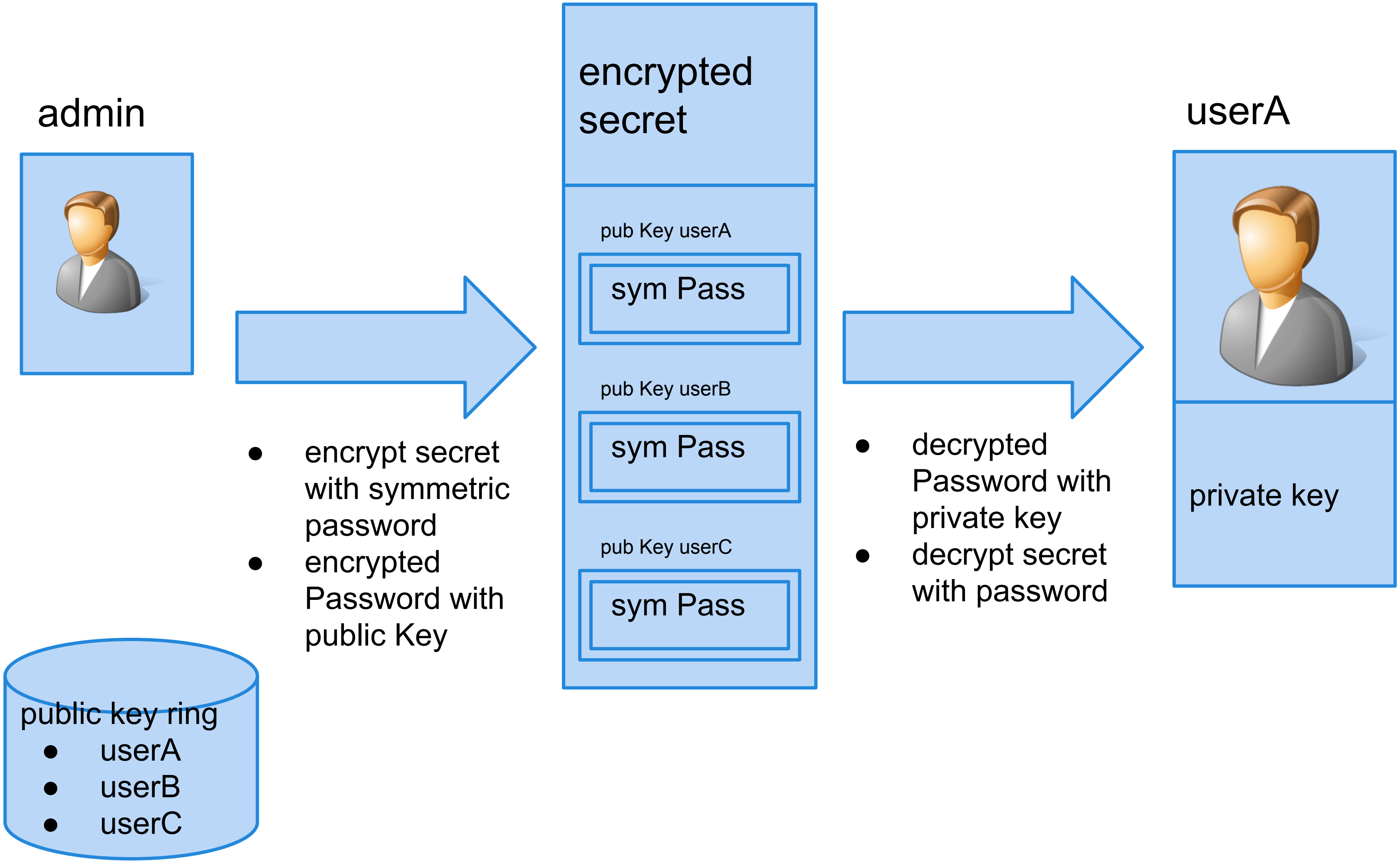
Tools

The current DESY tool chain

Escrow

- Manage passwords
- simple perl wrapper around pgp/gpg
- need user pgp/gpg keypair
  - painful
- secrets are readable by a defined set of users

Escrow



admin

- encrypt secret with symmetric password
- encrypted Password with public Key

public key ring

- userA
- userB
- userC

encrypted secret

pub Key userA

sym Pass

pub Key userB

sym Pass

pub Key userC

sym Pass

- decrypted Password with private key
- decrypt secret with password

userA

private key

Escrow

+ secure

+ simple to setup and extend

- no management/control/deploy

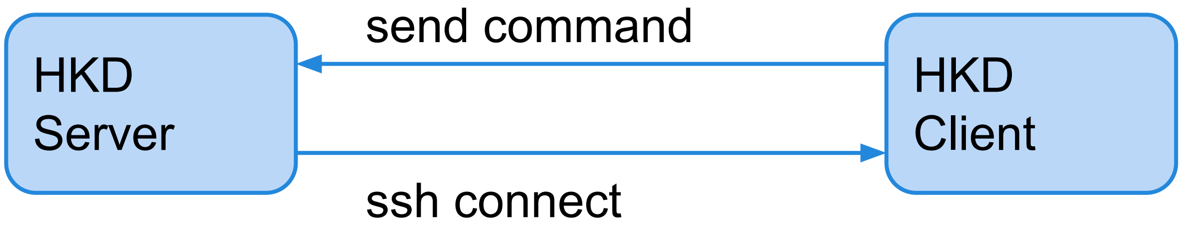
- no groups

- need shared filesystem

HKD

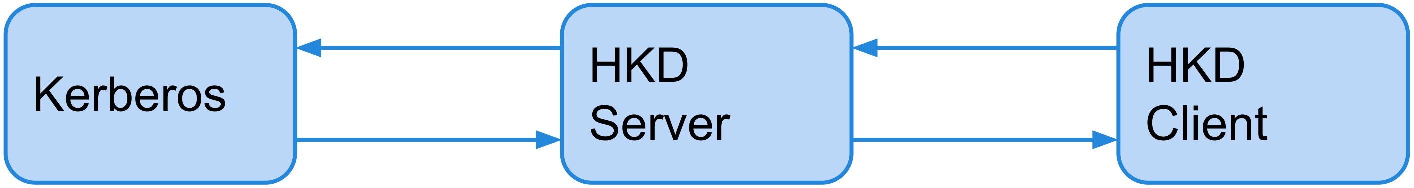
- Save and restore secrets
- Create kerberos principals
- Control status of secret
- Simple client/server shell script

HKD



1. Client add HKD ssh-key to allow root login
2. Client send command and hostname to server
3. Server connect via ssh to client and restore or save secret
4. Server send result
5. Client remove ssh-key

HKD



1. Client add HKD ssh-key to allow root login
2. Client send command and hostname to server
3. Server create kerberos principal
4. Server connect via ssh to client and store keytab
5. Server send result
6. Client remove ssh-key

HKD

+ secure in controlled networks

+ simple to setup and extend

- no management/control/deploy

- no groups

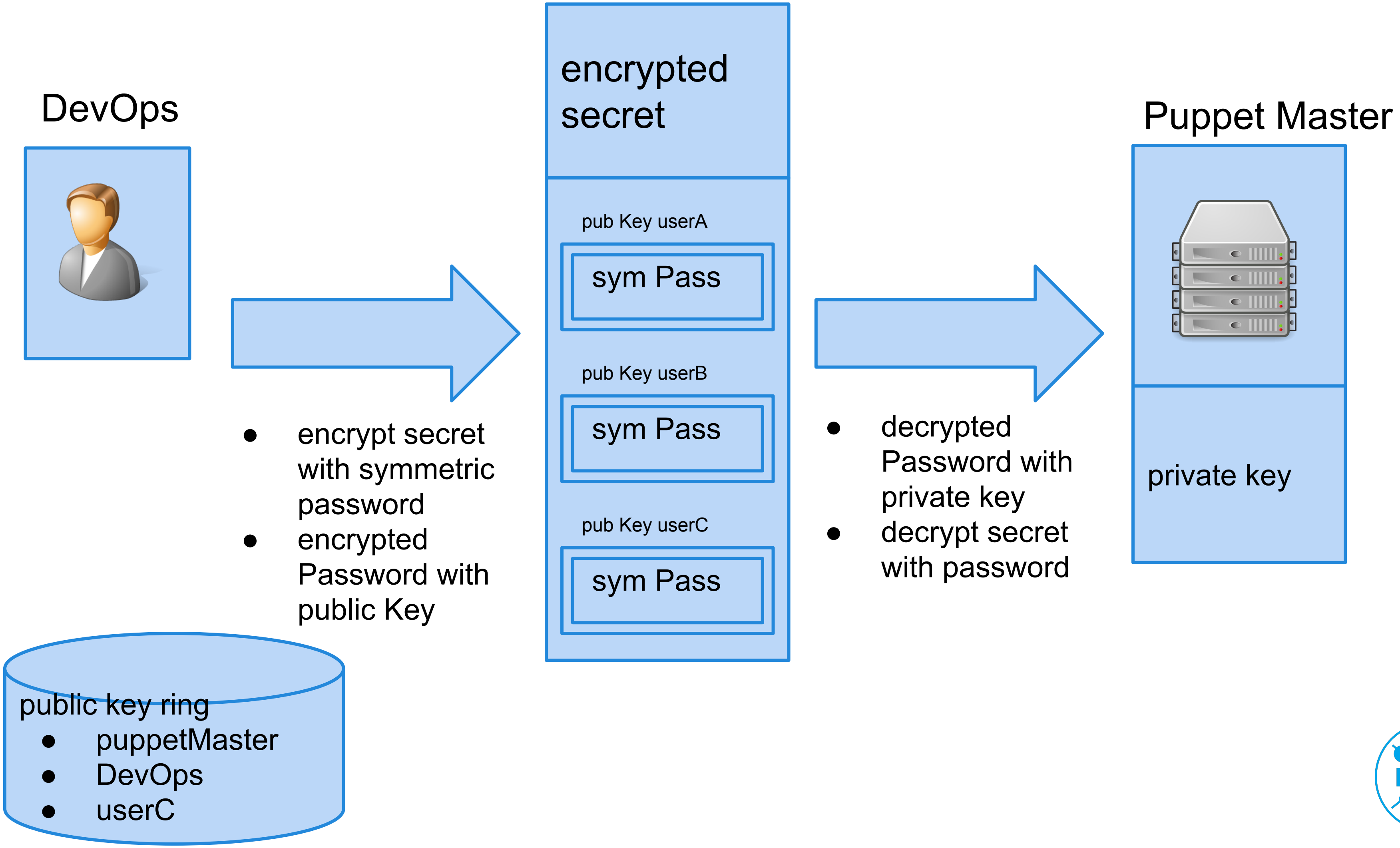
Tools

The new DESY tool chain

Encrypted puppet/hiera

- Store secrets in hiera
- Access secrets via puppet/hiera
- Encrypted secret
  - puppetmaster can read
  - DevOps can read and write
- Doesn’t work in an agile/DESY world

Encrypted puppet/hiera



DevOps

- encrypt secret with symmetric password
- encrypted Password with public Key

public key ring

- puppetMaster
- DevOps
- userC

encrypted secret

pub Key userA

sym Pass

pub Key userB

sym Pass

pub Key userC

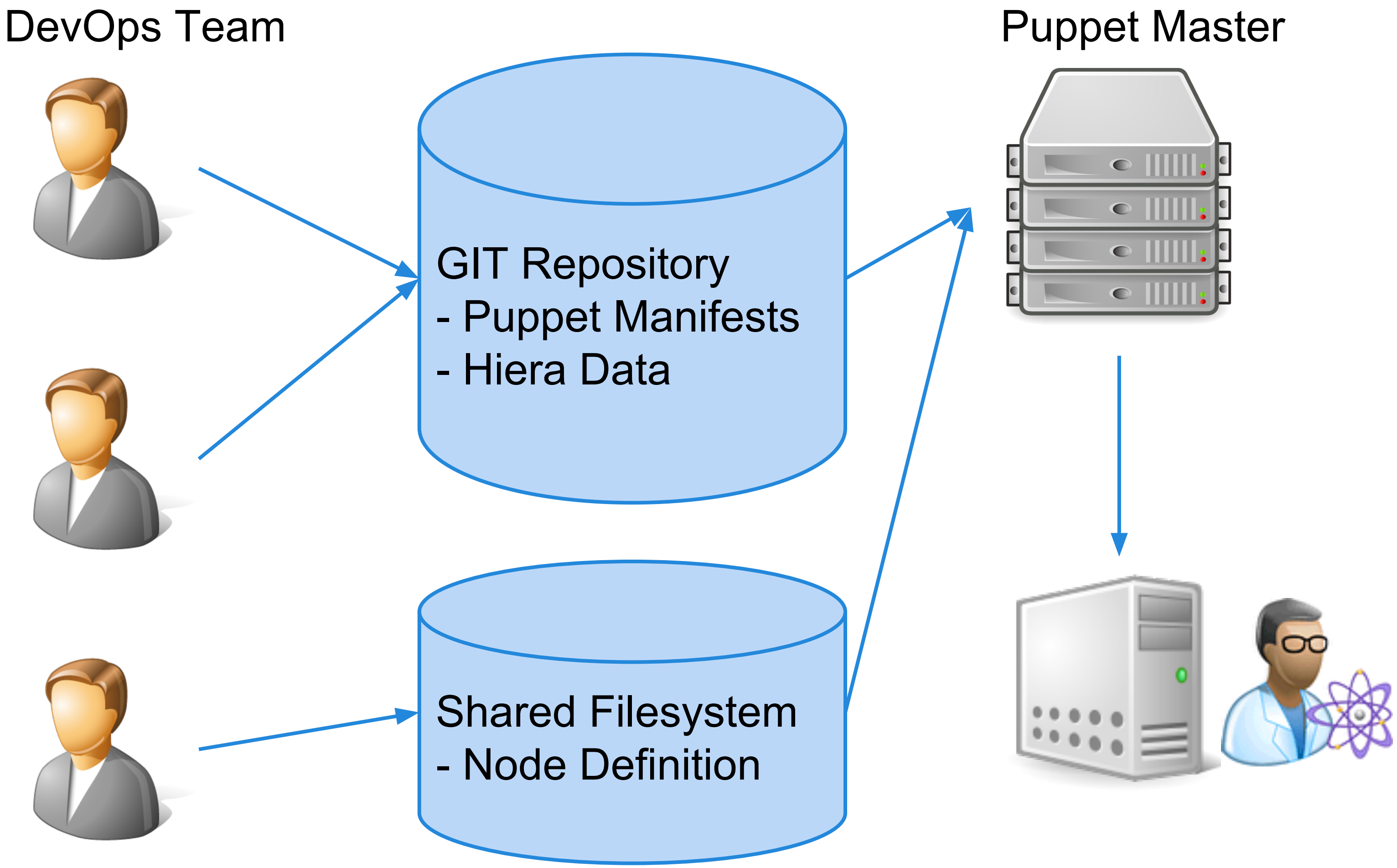
sym Pass

- decrypted Password with private key
- decrypt secret with password

Puppet Master

private key

Encrypted puppet/hiera



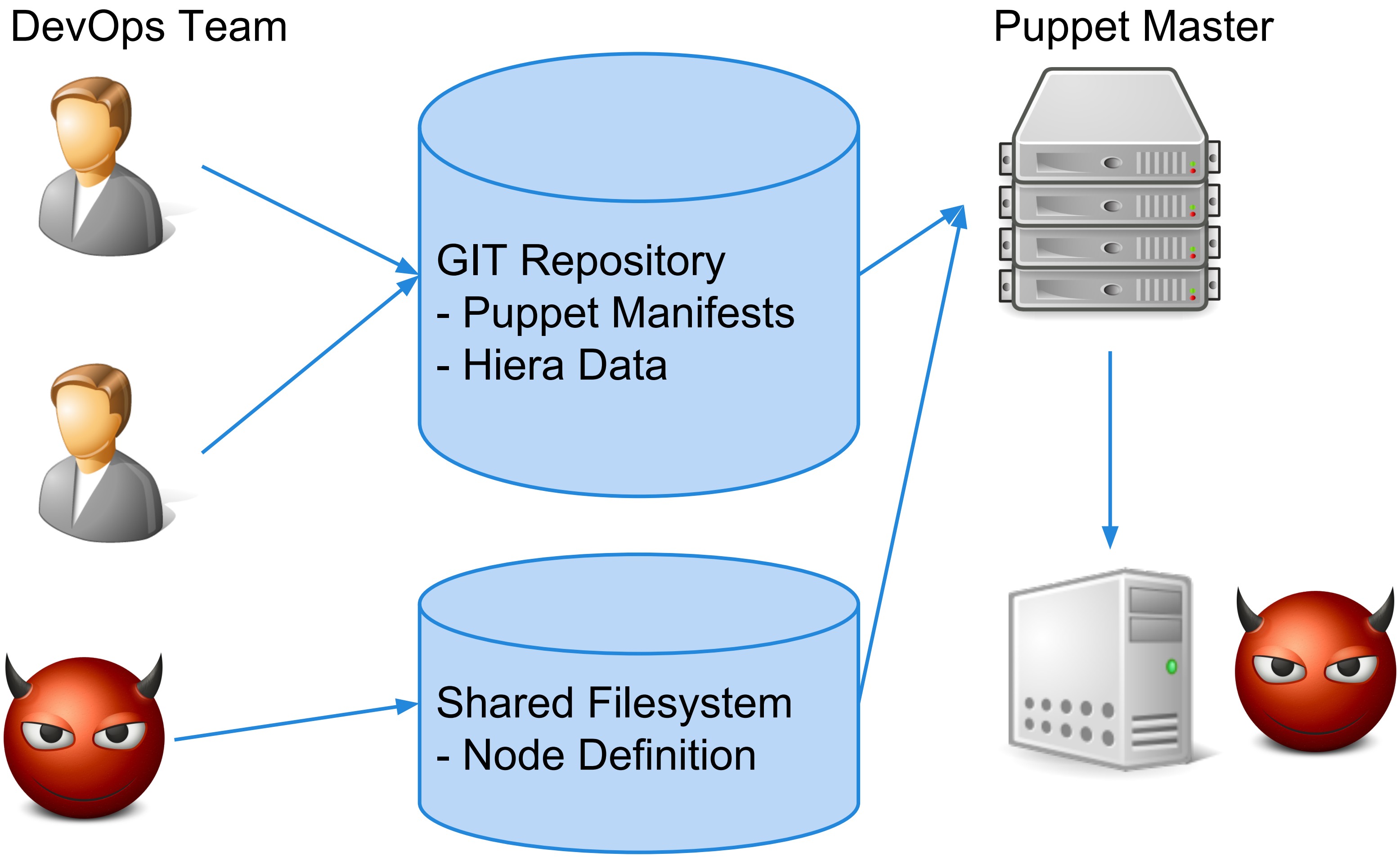
DevOps Team

GIT Repository
- Puppet Manifests
- Hiera Data

Shared Filesystem
- Node Definition

Puppet Master

Encrypted puppet/hiera



DevOps Team

Puppet Master

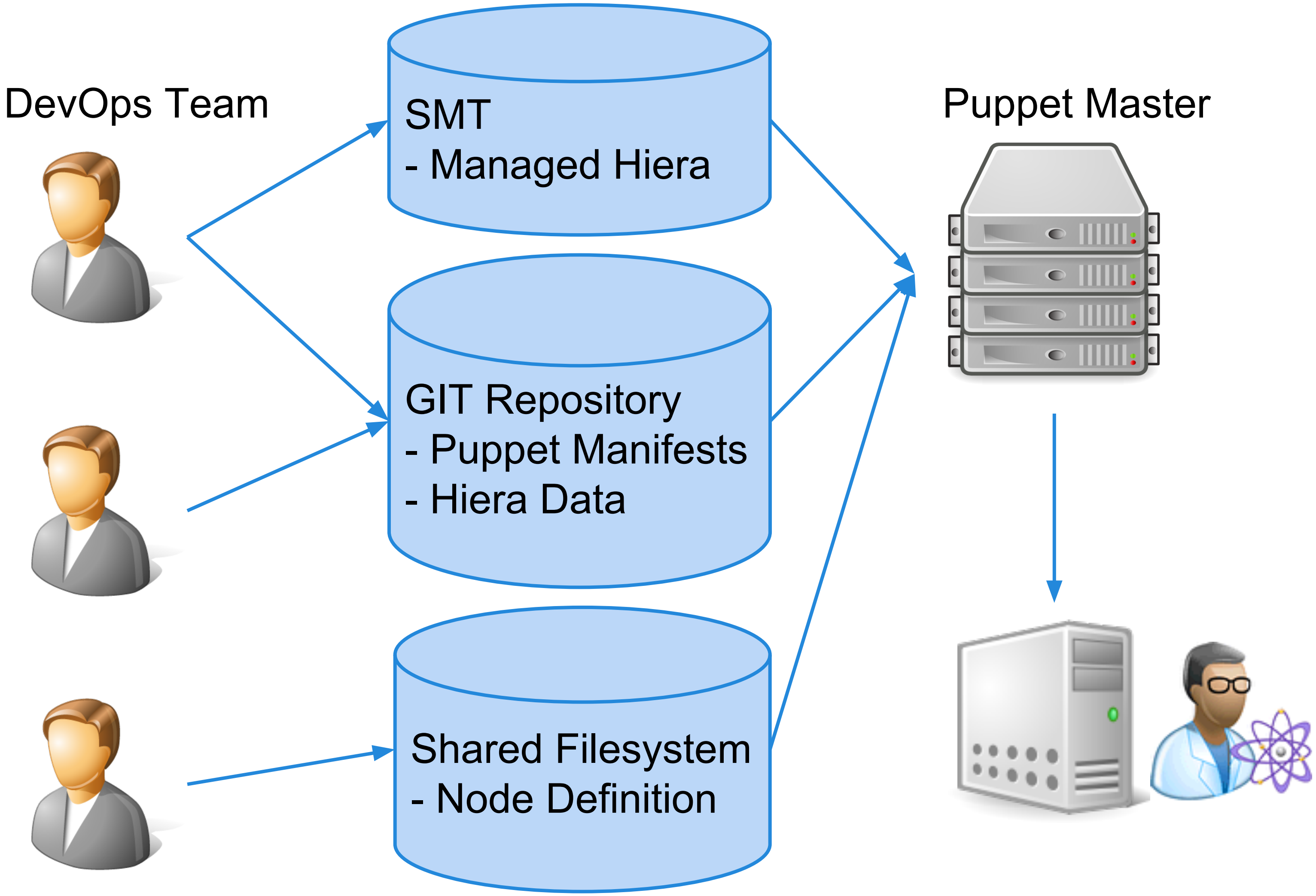
GIT Repository
- Puppet Manifests
- Hiera Data

Shared Filesystem
- Node Definition

Secure Management Tool

- Prototype developed in master thesis of Holger Bischof / HTW Berlin
- Store secrets in separate DB
- REST Interface
  - secrets accessible via hiera/puppet
- ACL for user and node
- Not only useful for secrets in Puppet environments

Secure Management Tool



DevOps Team

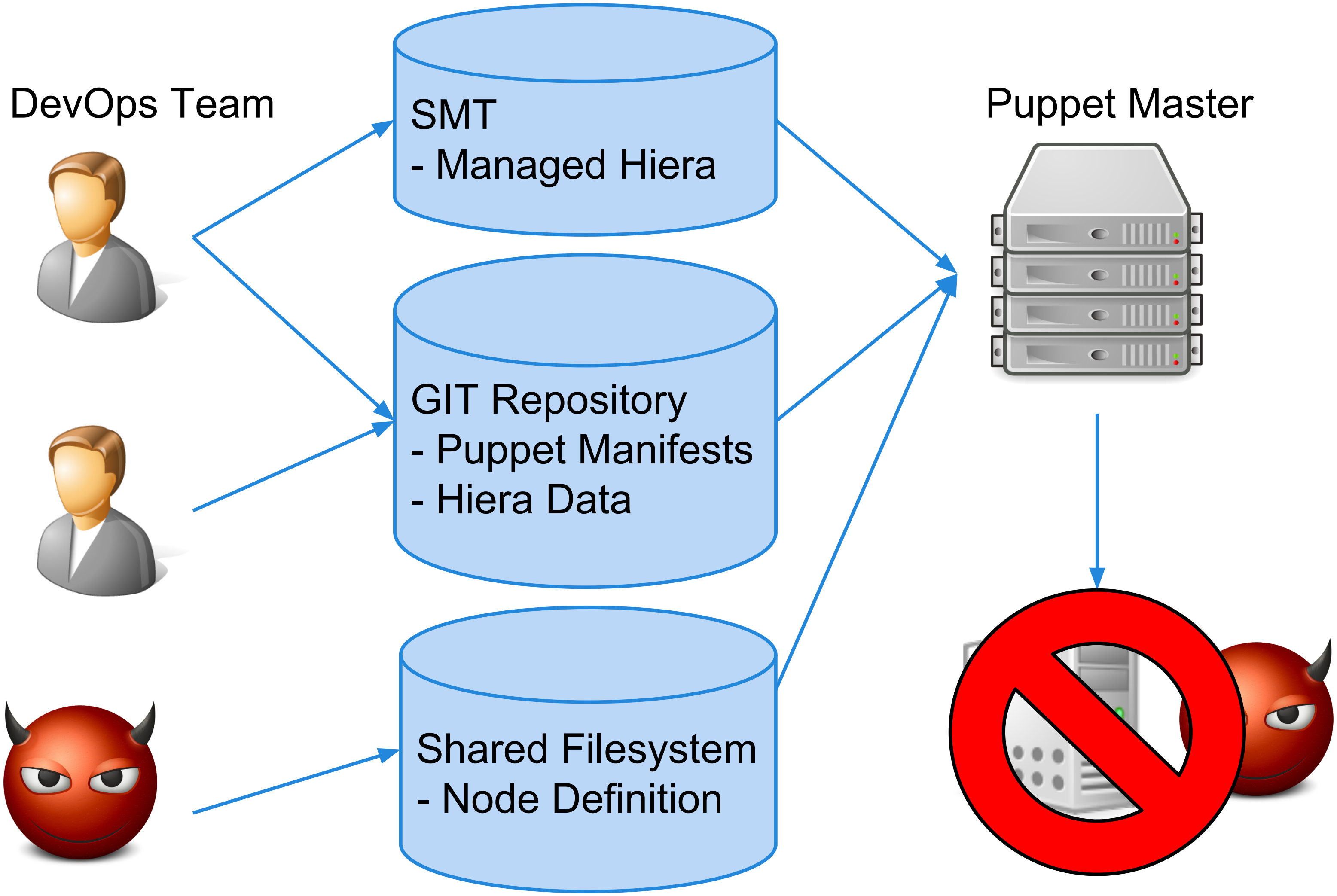
SMT
- Managed Hiera

GIT Repository
- Puppet Manifests
- Hiera Data

Shared Filesystem
- Node Definition

Puppet Master

Secure Management Tool



DevOps Team

SMT
- Managed Hiera

GIT Repository
- Puppet Manifests
- Hiera Data

Shared Filesystem
- Node Definition

Puppet Master

Secure Management Tool

+ together with puppet good management capabilities

+ could be used without puppet

+ flexible

- complex software

- Puppet not initially developed for secrets management

Secure Management Tool

Puppet security shortcoming, probably more than this but:

```
$master_cert = file('/var/lib/puppet/ssl/private_keys/<FQDN>.pem')
```

`see http://docs.puppetlabs.com/references/stable/function.html#file`

The End

Q & A