openstack.

May 23, 2018

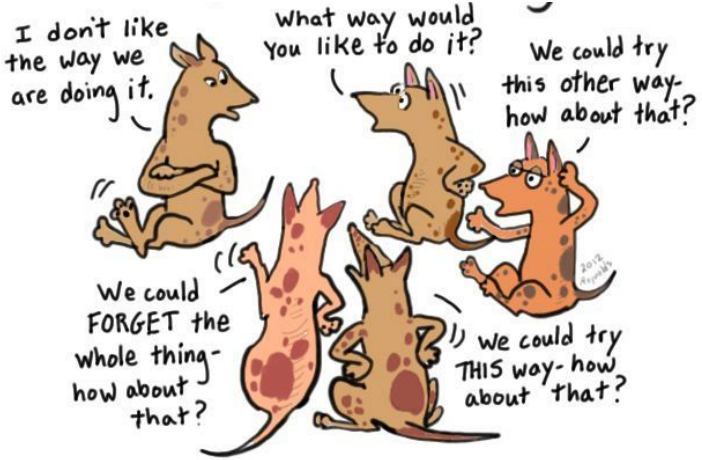
Tatu:
Better SSH management for clouds

OpenStack Summit Vancouver

Pino de Candia, @pino_deca

Some problems with SSH in OpenStack

1. MITM vulnerability on first connection.
2. No automated way to revoke user access.
3. Need a FloatingIP per instance (or manage bastions).
4. No integration with Keystone users, roles, etc.



Problem 1: MITM vulnerability

```
ECDSA key fingerprint is….
Are you sure you want to continue connecting (yes/no)?
```

Does anyone check the fingerprint?

How would we anyway? The key is generated on first boot.

Problem 2: Public key management

OpenStack writes the user public key to the instance's authorized_keys file. It's better than password auth.

But what about access for multiple users?

- Share the private key?
- Add more keys to authorized_keys file… but who cleans up?

Problem 3: FloatingIP per instance

- Wastes FloatingIP addresses.
- Use some instances as bastions?
  - Who manages the bastions and how?

Problem 4: No integration with Keystone

openstack.

Separate Identity Management for platform and servers.

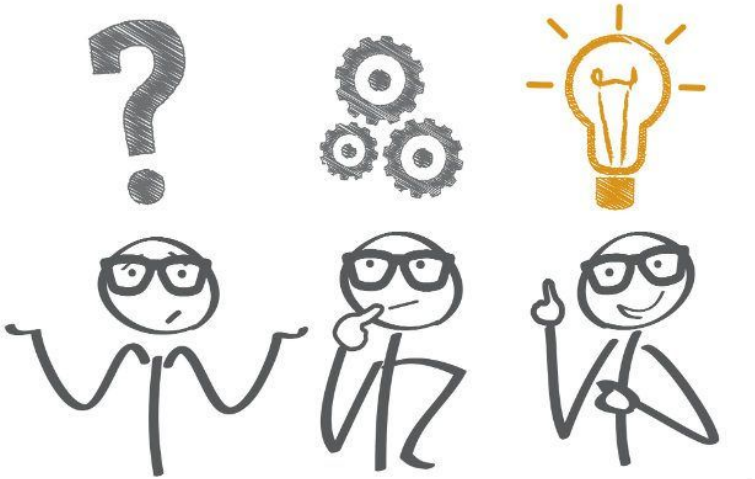
OpenStack roles don’t correspond to server logins/accounts.

Changes in user permissions don’t change SSH access.

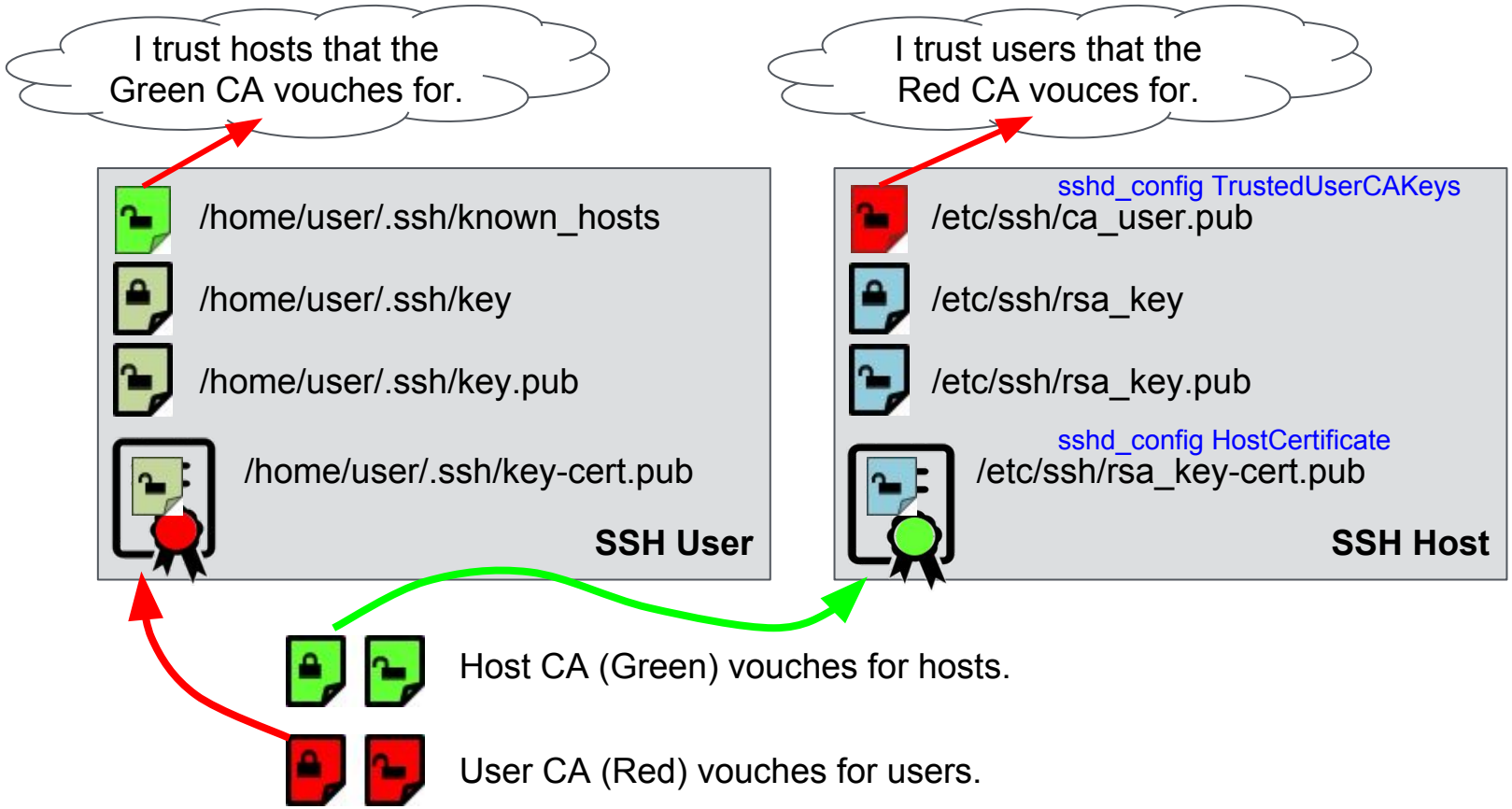
(Partial) Solution: SSH Certificates

openstack.

- Available since OpenSSH 5.4
- For both Users and Hosts
- SSH clients trust the Certificate Authority, not the hosts.
- Similarly, SSH hosts trust the CA, not the users.



SSH with Certificates



So, what's Tatu? What are its goals?

- Ease adoption of SSH Certificates in OpenStack
- New OpenStack CLI and Horizon panels for users to:
  - Discover the Host CA public key
  - Generate and revoke user certificates
- Automate the SSH setup on instances
- Automate server account setup based on Keystone roles
- Changes to users/roles result in SSH access changes
- Manages SSH bastions and DNS for ease of use

Tatu Project Status

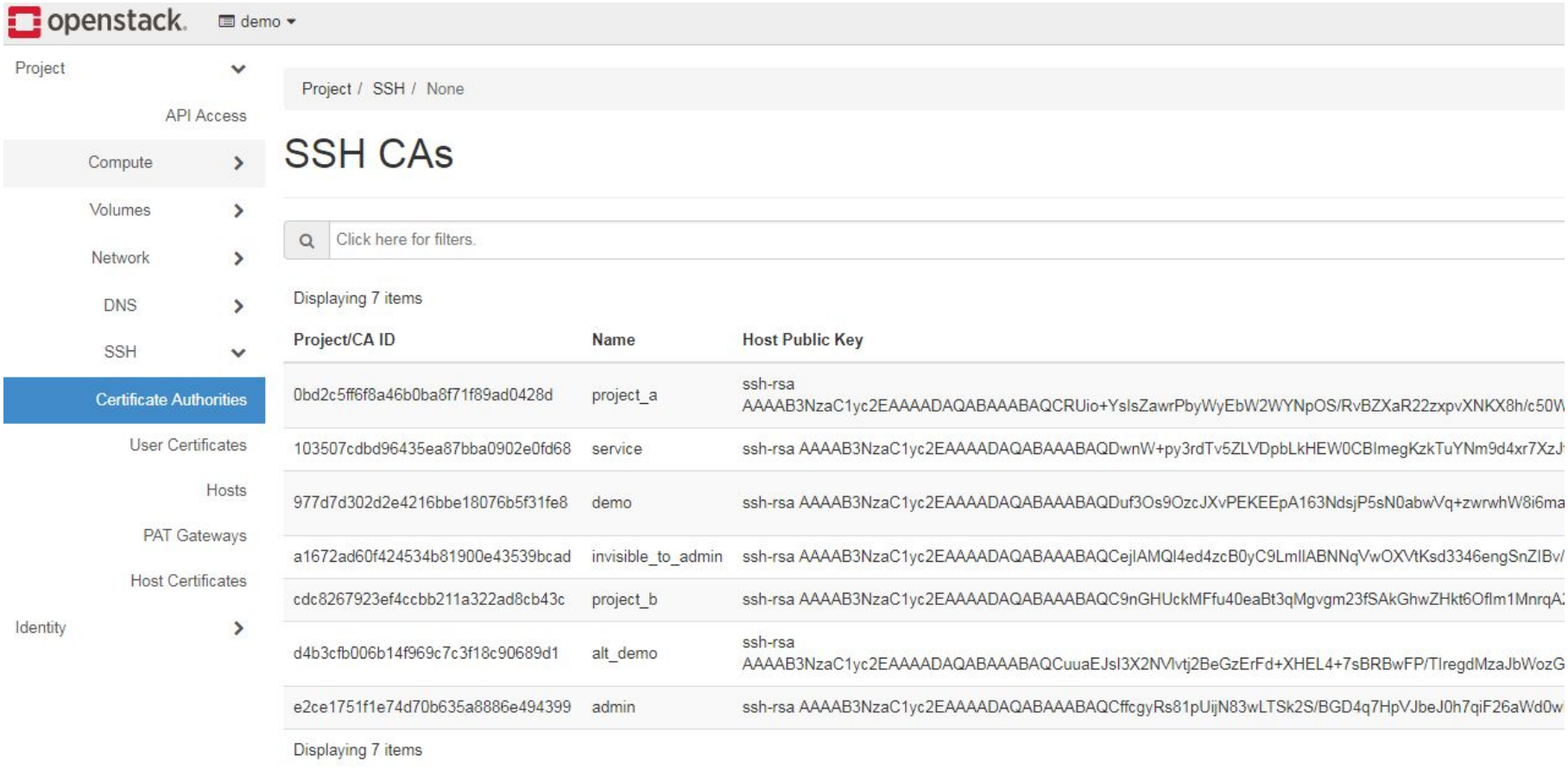
openstack.

- All code in OpenStack Github/Gerrit since early March.
- API server, CLI and Horizon panels completed
  - Caveat: panels need result filtering by project and user
- Completed Devstack plugin
- Experimental bastion support with Dragonflow
- Experimental pam-ussh integration restricts sudo in open sessions
- Experimental relationship between roles and server accounts

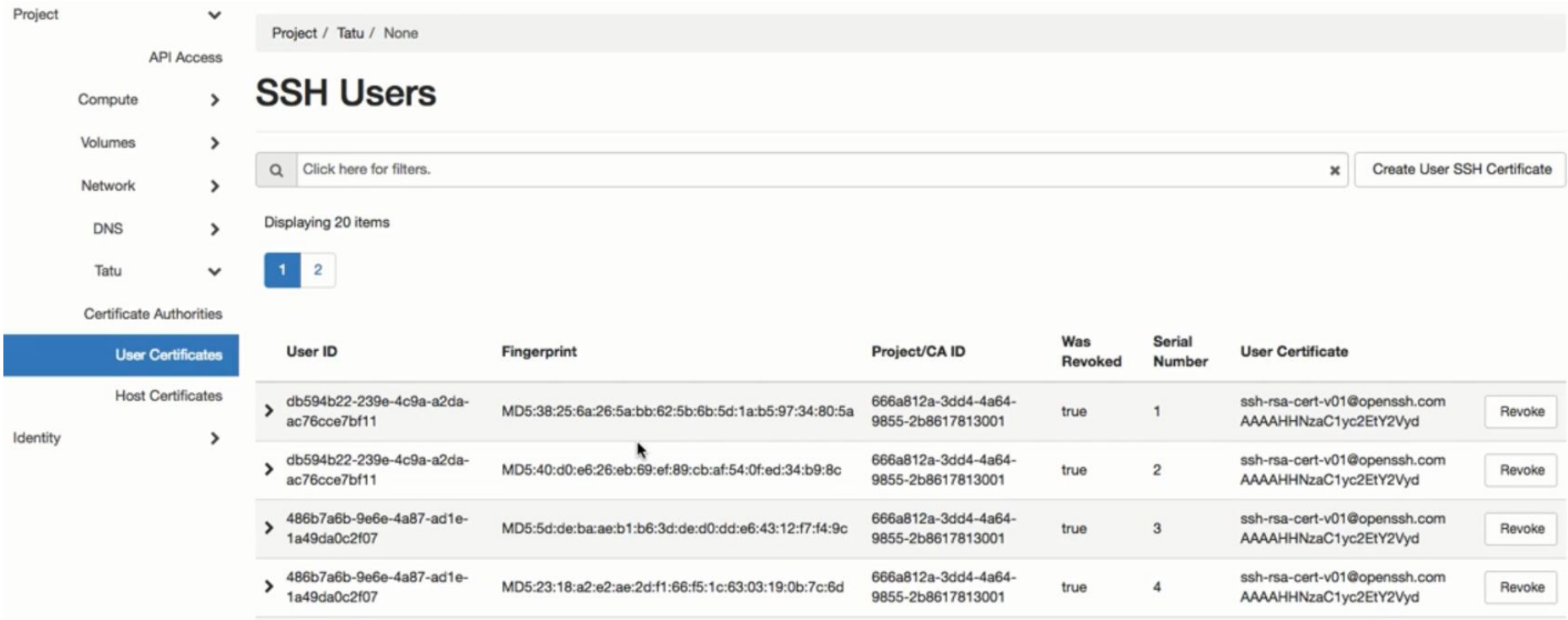
Not ready for production: needs deployment tools, CI, and more tests.

Looking for contributors and users!

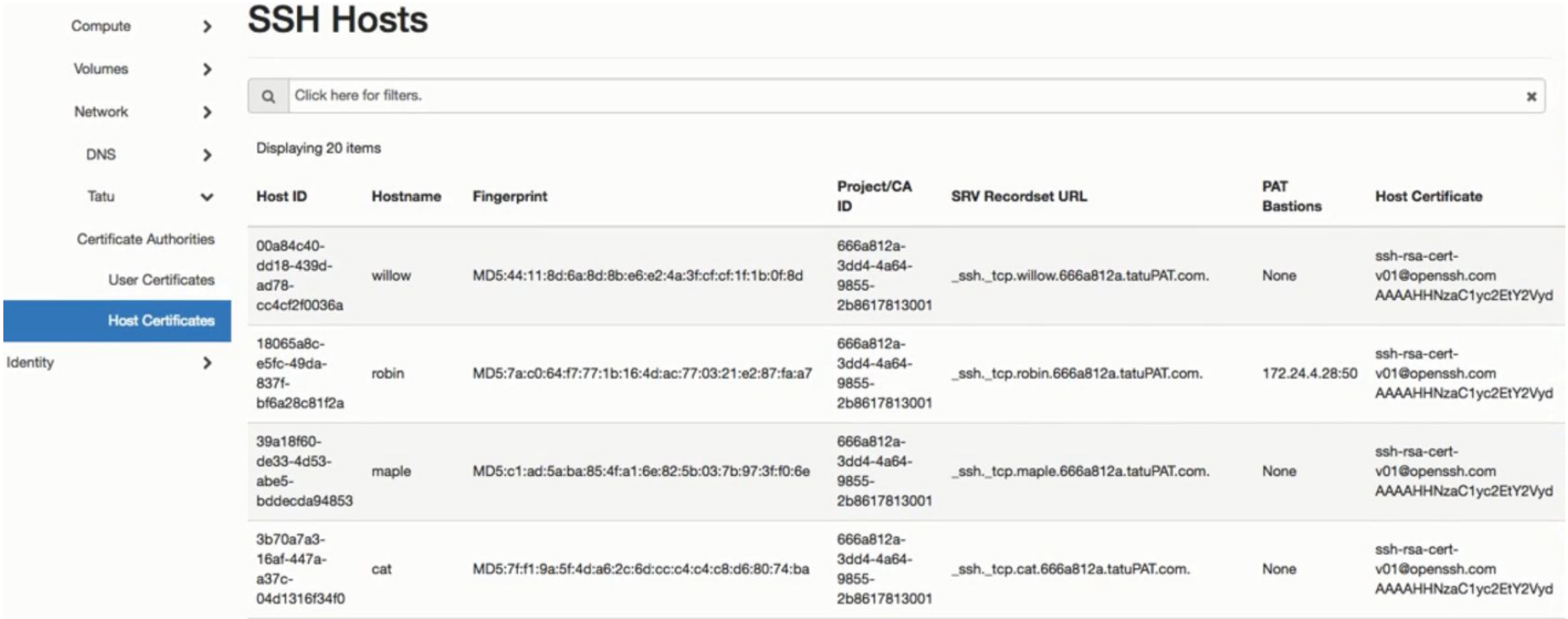
Horizon Panel - SSH Certificate Authorities



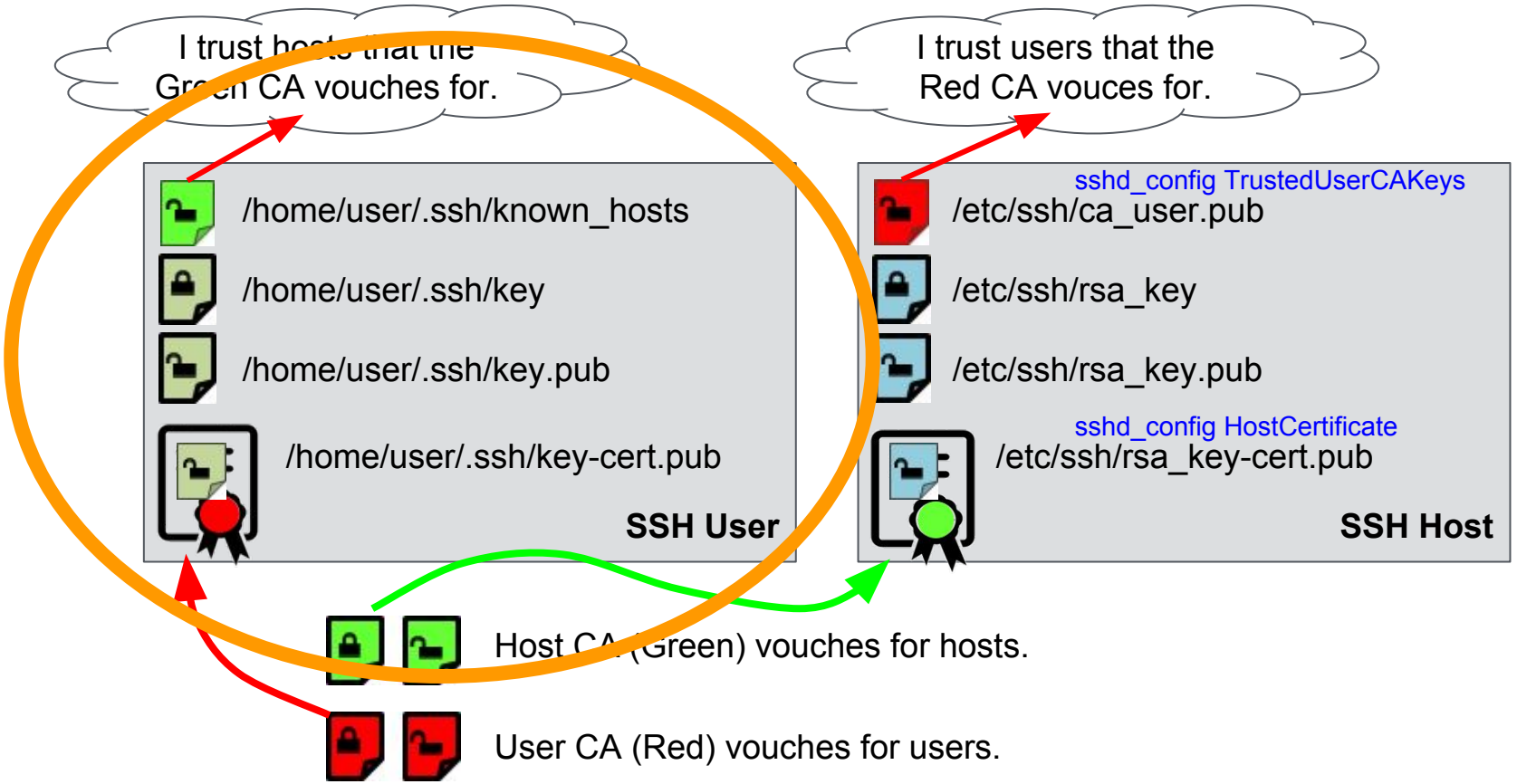
Horizon Panel: User Certificates



Horizon Panel: Host Certificates



User SSH setup



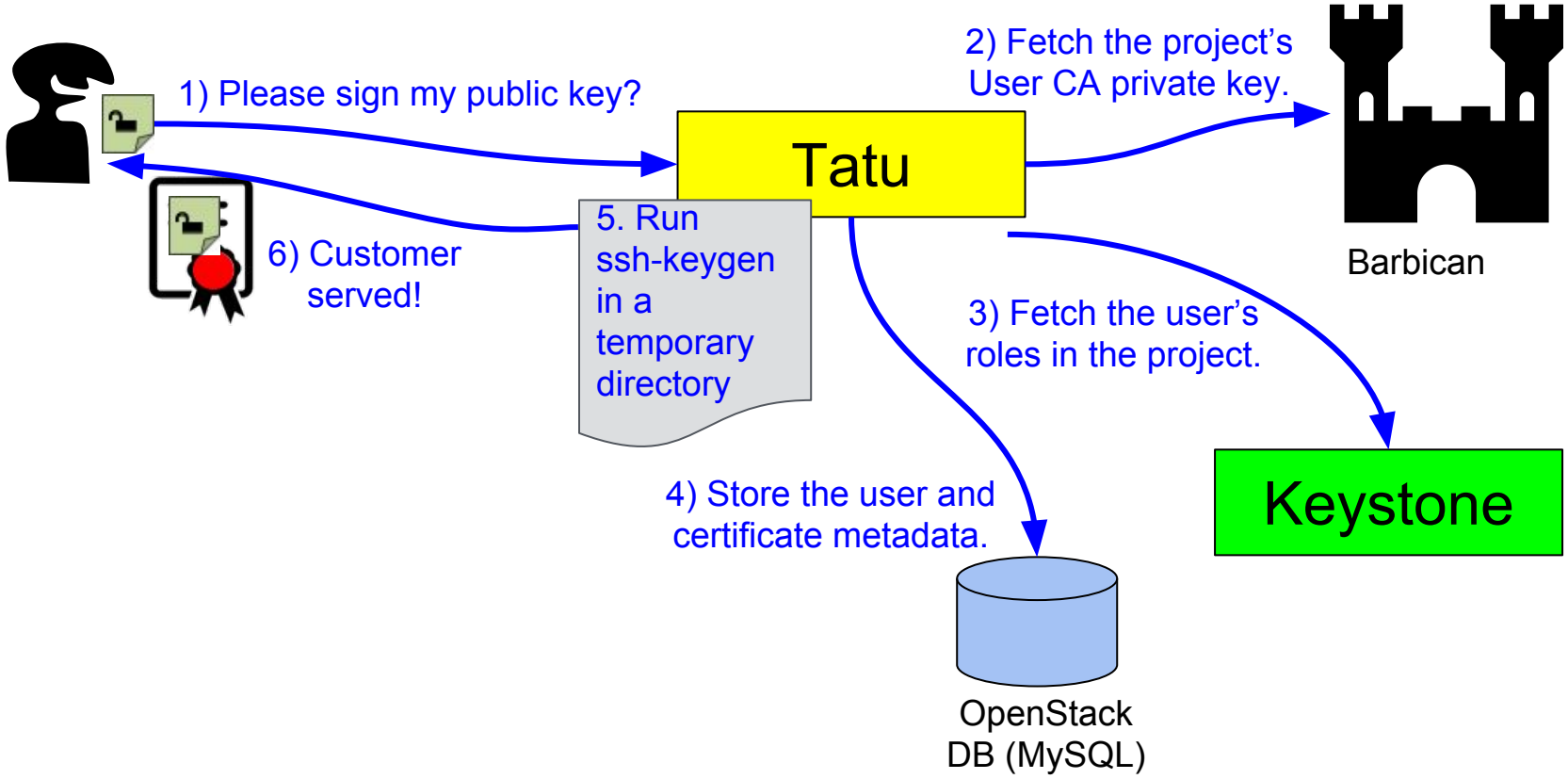
User SSH setup (once per project)

```
# These commands have been shortened for clarity (do not copy/paste)
echo '@cert-authority * '`openstack ssh ca show >> known_hosts

ssh-keygen -f demo_key

ssh-add demo_key

openstack ssh usercert create demo_key.pub > demo_key-cert.pub

ssh -A -v <keystone-role>@<address>

 ...
 debug1: Host XXX is known and matches the RSA-CERT host certificate.
 ...
 debug1: Offering RSA-CERT public key: demo_key-cert
 debug1: Server accepts key: pkalg ssh-rsa-cert-v01@openssh.com blen
1088
```

User SSH setup (once per project)

```
# Commands have been shortened for clarity (do not copy/paste)
echo '@cert-authority * '`openstack ssh ca show >> known_hosts
ssh-keygen -f demo_key
ssh-add demo_key
openstack ssh usercert create demo_key.pub > demo_key-cert.pub
ssh -A -v <keystone-role>@<address>
 ...
 debug1: Host XXX is known and matches the RSA-CERT host certificate.
 ...
 debug1: Offering RSA-CERT public key: demo_key-cert
 debug1: Server accepts key: pkalg ssh-rsa-cert-v01@openssh.com blen
1088
```

User certificate generation



User certificate details

```
# ssh-keygen -Lf ~/.ssh/key22-cert.pub
/root/.ssh/key22-cert.pub:
    Type: ssh-rsa-cert-v01@openssh.com user certificate
    Public key: RSA-CERT SHA256:ZDsaPxjKHlBo6BWf3R0OKokrNU+T3TDy8MU5v1YOJAY
    Signing CA: RSA SHA256:LQ5ikXe8LybhCFiuWGuiVqagSIyy2eiYpRhhu9lWnfw
    Key ID: "demo_22"
    Serial: 22
    Valid: from 2018-02-08T18:01:56 to 2019-02-09T18:01:56
    Principals:
        admin
        Member
        anotherrole
    Critical Options: (none)
    Extensions:
        permit-X11-forwarding
        permit-agent-forwarding
        permit-port-forwarding
        permit-pty
        permit-user-rc
```

Use serial numbers for certificate revocation.
Key ID is user's name plus serial number.

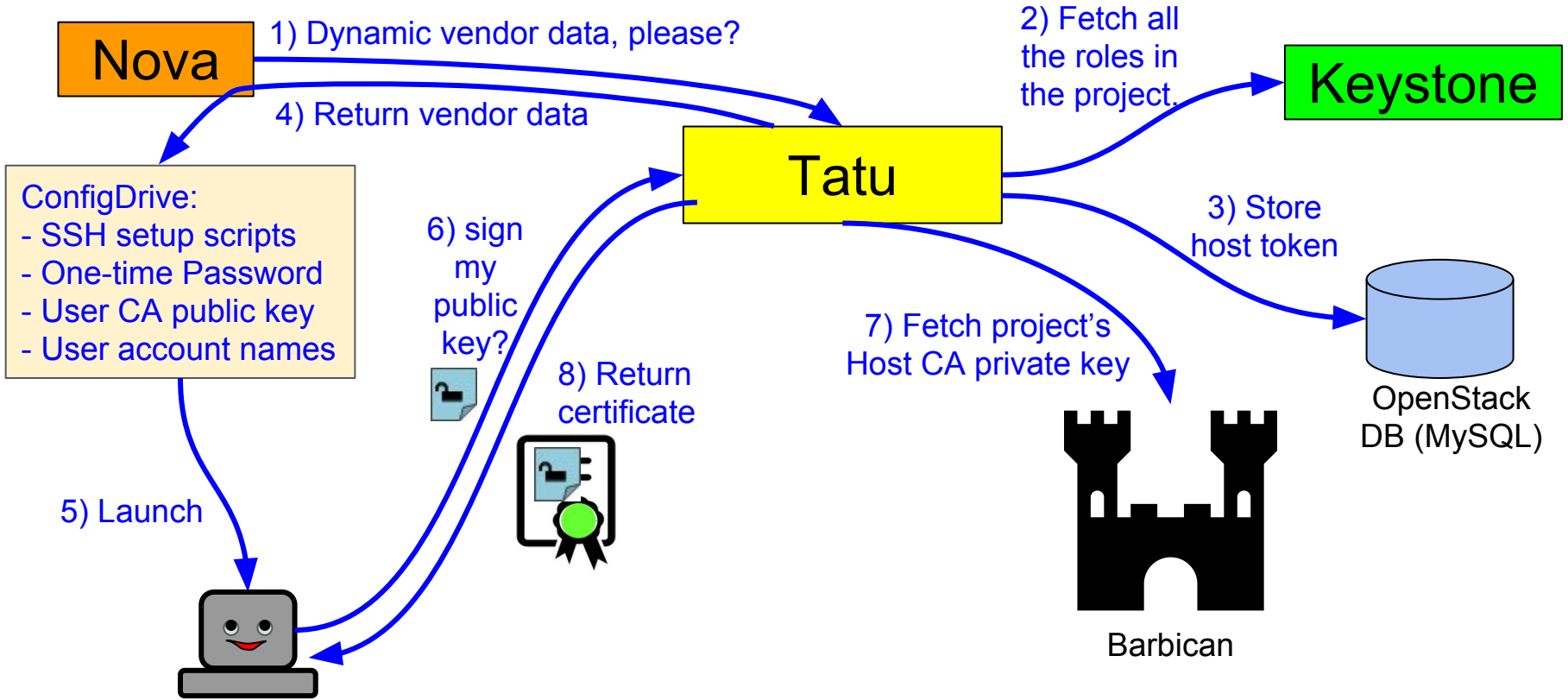
Set principals by querying Keystone for user's roles in project.

User's known_hosts file

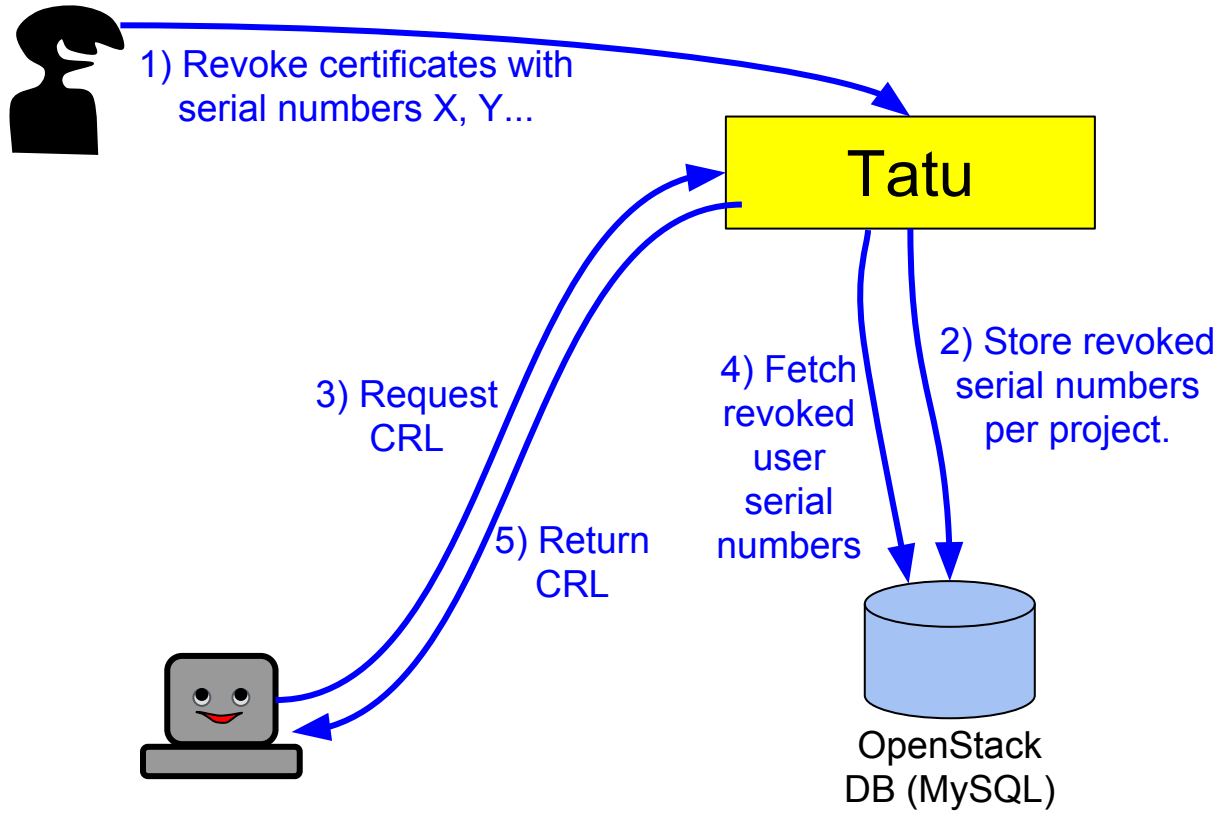
openstack.

```
@cert-authority * ssh-rsa
 AAAAB3NzaC1yc2EAAAADAQABAAABAQDmdIeP5Adh6sDc+IzvFYTo0W7dhVpK
 WgMvmOsXXm8VgGFBYJTLK/xLeXC8gIR+NNRs7xKYzPSBCc25Qdvm59rBCpd2
 WZUJ0peQ14r6PtW20Xc26iOy39UMcmJEP+QWTQ8yjOZvd4MbU892qQJt0CjD
 xR8ac+hXbRZ7e9zp8AJhwCkBDlZRkc1LWMHI0s4Lh5pCieJOrDTaJzdCyg9D
 KSWuxKSRT4OeGJe/2ELPJ3jL3YPi40KhxXlV3L9PpGpftAb7tLR20lhWXVbx
 Ud/D6u5aQFPXye91AQoXoaSWDYg6KSayLscwgHU2tgXa4Nb5HDWm5bNRdW0P
 70dS465X+0e3
```

New instance (host) certificate generation

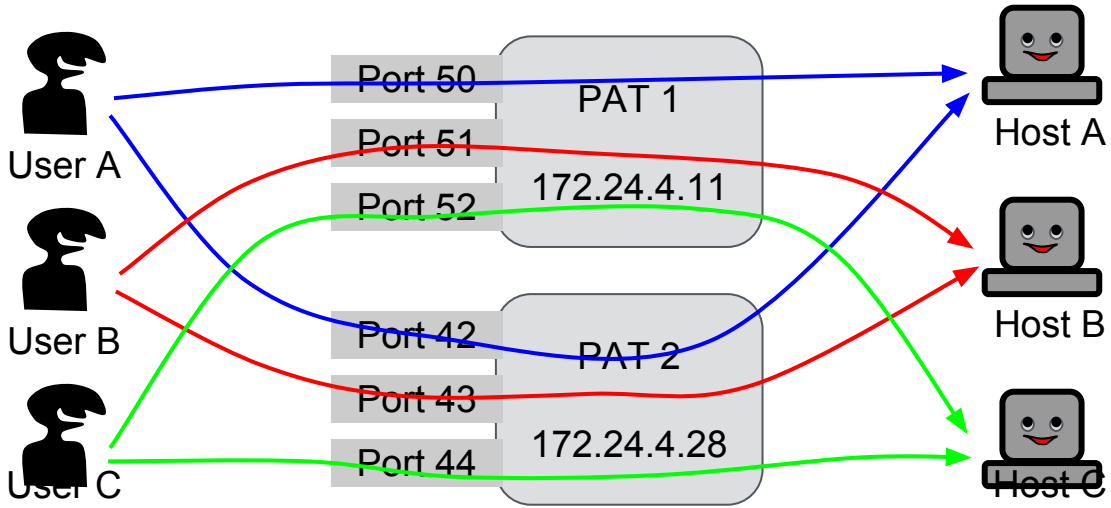


Instance periodically fetches CRL (cron job)



PAT (port address translation) bastions

- Real bastion VMs are better (planned feature).
- But PAT already allows us to avoid FloatingIP-per-instance.
- Like load-balancing: one VM gets one port on a few PAT proxies.
- Currently only works with Dragonflow.



Integration with Designate

- Users shouldn't have to remember bastion or instance IP addresses.
- They should just SSH to a URL composed of instance and project names.
- In the case of PAT bastions, we look up SRV records rather than A records.

```
### tatu/scripts/srvssh -i ~/.ssh/key root@myvm1.demo.tatuPAT.com.
srv: '_ssh._tcp.myvm1.demo.tatuPAT.com has SRV record 10 50 52
bastion-172-24-4-11.tatuPAT.com.'
srv: 'bastion-172-24-4-11.tatuPAT.com has address 172.24.4.11'
After SRV lookup -- HOST: 172.24.4.11 PORT: 52
/usr/bin/ssh -v -i /root/.ssh/key22 -p 52 root@172.24.4.11
```

In the example, the instance name is 'myvm1'; project name is demo;
srvssh is a wrapper script that resolves the SRV record and then calls ssh.

Successful use of certificates looks like...

```
/usr/bin/ssh -v -i /root/.ssh/key22 -p 52 root@172.24.4.11
OpenSSH_7.2p2 Ubuntu-4ubuntu2.4, OpenSSL 1.0.2g  1 Mar 2016
...
debug1: Server host certificate: ssh-rsa-cert-v01@openssh.com
 SHA256:tobg2ZPvX45OpqQh9dHlRiZxkg9Vmv6gUQaV1T9dyss, serial 0 ID "testID" CA
 ssh-rsa SHA256:qS95+uo3KEzwxyrihHfe4NWqKTUX64bRSfCWuG5i5RM valid from
 2018-02-07T18:26:19 to 2019-02-08T18:26:19
debug1: Host '[172.24.4.11]:52' is known and matches the RSA-CERT host
 certificate.
debug1: Found CA key in /root/.ssh/known_hosts:1
...
debug1: Offering RSA public key: /root/.ssh/key22
debug1: Authentications that can continue:
 publickey,gssapi-keyex,gssapi-with-mic,password
debug1: Offering RSA-CERT public key: /root/.ssh/key22-cert
debug1: Server accepts key: pkalg ssh-rsa-cert-v01@openssh.com blen 1095
...
```

Client accepted host cert

Server rejected user's public key since it's not in the authorized_keys file.

Host accepted user cert

In the background...

A Tatu daemon reacts to these oslo event notifications:

- Project creation: create new CA key pairs.
- User deletion: revoke all the user's certificates.
- Deletion of role assignment: revoke certificates that grant access to the corresponding accounts on servers.
- Host deletion: clean up PAT port entries in Dragonflow and DNS entries in Designate.

Summary: SSH current user experience

openstack.

1. Generate SSH key pair (locally with ssh-keygen)
2. Upload public key to OpenStack
3. Launch VM. Include key pair and assign Floating IP
4. SSH into VM
    - **`ECDSA key fingerprint is…. Are you sure you want to continue connecting (yes/no)?`**
    - How do I verify the fingerprint? I'll just cross my fingers.
5. When multiple colleagues want access to same VMs, share private keys or add public keys to authorized_keys file.
6. When colleagues leave wait weeks or months before removing their public keys or rotate shared keys.

Summary: SSH user experience with Tatu

openstack.

1. Generate SSH key pair (locally with ssh-keygen)
2. Get your key signed by your project's CA
   - /home/pino/.ssh/key-cert.pub
3. Put the CA's **host** public key in the known_hosts file
   - `@cert-authority <domain> ssh-rsa AAAAB3NzaC…`
4. Launch VM **without key pair and without Floating IP.**
5. SSH into VM via automatically assigned PAT IP+port and without MITM risk.
6. Colleagues automatically have access without sharing keys.
7. Access can be revoked with a click and is automatically revoked if the user is deleted.

Reference commands

```
host -t SRV _ssh._tcp.<hostname>.<project_id>.tatuPAT.com. localhost

dig @localhost -t SRV _ssh._tcp.fox.666a812a.tatuPAT.com.

/opt/stack/tatu/scripts/srvssh -i ~/.ssh/key root@fox.666a812a.tatuPAT.com.

ssh -i ~/.ssh/key -p 40 root@172.24.4.11

openstack ssh ca show -f value -c 'Host Public Key' <project_id>

echo '@cert-authority *' `openstack ssh ca show -f value -c 'Host Public Key'
666a812a-3dd4-4a64-9855-2b8617813001` > /root/.ssh/known_hosts

openstack ssh usercert create -f value -c Certificate "`cat
~/.ssh/key14.pub`" > ~/.ssh/key14-cert.pub

openstack ssh usercert revoke <serial number>
```

Potential future work

- Bastion VM management
- Centralized SSH audit logs
- Use Uber's PAM module to validate existing SSH sessions
  - https://github.com/uber/pam-ussh
  - In-progress
- Rotation of CA keys
- Rotation of host certificates
- Installers
- SSH client integrated with Single-Sign-On

Ideas are welcome!

Resources

- https://github.com/openstack/tatu
- https://github.com/openstack/tatu-dashboard
- https://github.com/openstack/python-tatuclient
- https://launchpad.net/tatu

Demo Videos:

- Full: https://youtu.be/y6ICCPO08d8
- Disable sudo on existing connections (uses Uber's pam-ussh): https://youtu.be/yjwWdYJRTM0

IRC: #openstack-tatu on freenode.net

openstack

Q&A

Thank you!

 openstack

 @OpenStack

 openstack

 OpenStackFoundation