

Jeremy Stott jeremy@stott.co.nz @jsstott

Secure Shell (SSH)

What is it?

As a system administrator, I need to connect to a server remotely to...

- * apply an update
- * deploy an application
- * install a new package

\$ sudo adduser jeremy

\$ ssh usbarmory@localhost



Puppet

\$ sudo puppet agent --enable && \
 sudo puppet agent -t --noop ; \
 sudo puppet agent --disable

- 0

Public Keys

```
$ ssh-keygen -f owasp
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
```

UGgrcvWSNs jeremy@jeremy-laptop

The key's randomart image is:

+---[RSA 3072]---+

=*Xo=

+.++B..

+ *.= .

cat owasp.pub ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDJ VluyDrjz9T4Bpwwe2FGQAJ8wJ197LZm4uvZk/58w TeXaNNSMGfEE07J9+dxDDRC5kwc7S+K4HyTEnIJK fNkFBYZf4zl6RjkTiCxkM/Rn2NsrFGyH8nVQqFsI 9er0pwp9Un2ijPssLZ4oE1He4u6KieyF0gRgoLbX k6eZGh1uZNu3AFarUEQILKGCLXYtQsssKICYRxcH

ElFooHL7d0nJi/uyDoGYxJCMV2cqGc1nU3IcIKDk 19rwd43RJt1IwH/w3Vqcb3U3C50ySUNiowLEexMZ udwfnBvNVkhATJN+Yd6VtdRvV/c9nrm0G3amygGu vG+Lh6o03+dvxnxXM4C7b2a94G4UfUzv4SLw7I0H

u0o72hDRRGvwv9tXsHgREsIzWxNPwQjMl4i1VKl0 HFT8oi1H7BGB4DS0SD1gwv6Vr+UZLCzMVXNEfbvR QnDLorUISXcvq4Ye3A3ZggxR1HFSUjU= jeremy@ jeremy-laptop

This bit is public, put it on the server ~/.ssh/authorized keys

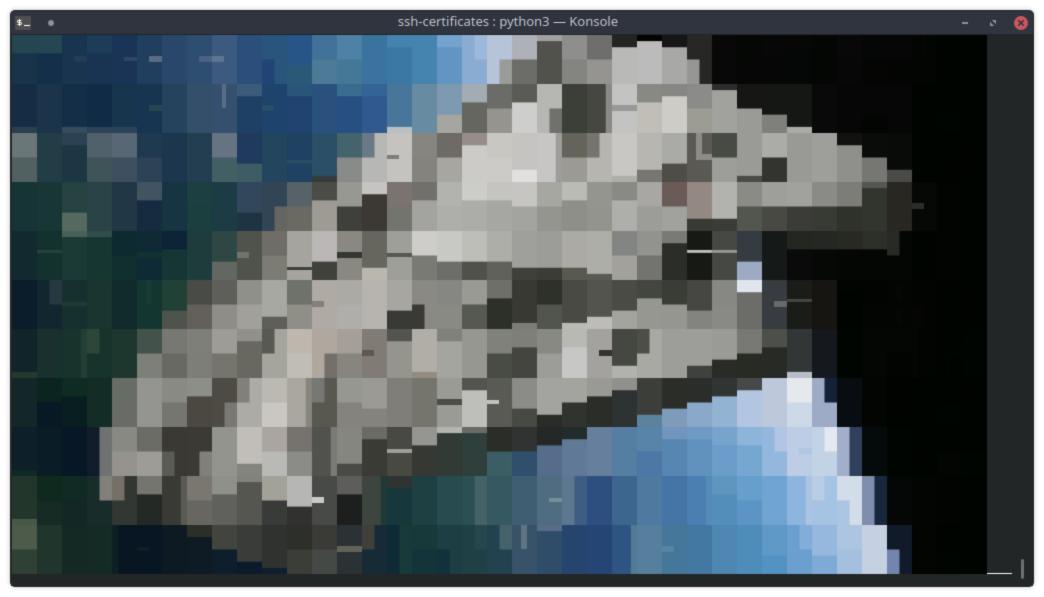
ssh ubuntu@server

cat owasp.pub

This bit is public, put it on the server ~/.ssh/authorized_keys

ssh ubuntu@server

/etc/ssh/sshd_config PasswordAuthentication no



As a support agent...

I need to SSH to a server and run random SQL queries against the db

SSH Certificates

public key + metadata + signature

SSH Certificates

- \$ ssh-keygen -f user
- \$ ssh-keygen -f ca
- \$ ssh-keygen -s ca -I id -V +1d -n haley
 user.pub

- 💲 ssh-keygen -f user
- \$ ssh-keygen -f ca
- \$ ssh-keygen -s ca -I id -V +1d -n haley user.pub Signed user key user-cert.pub: id "id" s
- erial 0 for haley valid from 2020-03-12T 15:05:00 to 2020-03-13T15:06:21

\$ ssh-keygen -f ca

\$ ssh-keygen -s ca -I id -V +1d -n haley
user.pub
Signed user key user-cert.pub: id "id" s

erial 0 for haley valid from 2020-03-12T 15:05:00 to 2020-03-13T15:06:21

\$ ls user*
user user-cert.pub user.pub

ssh-keygen -L -f user-cert.pub

user-cert.pub:

Type: ssh-rsa-cert-v01@openssh.c lom user certificate

Public key: RSA-CERT SHA256:sEdm

2ygxjPKVVmA4bnP+kx9TCylGafBd3PegjPgW0zg Signing CA: RSA SHA256: 15BdIjl0o QY6JhmkUuU3QmxbnNIfyzqxuxKbwpHvAd0 (usin

ssh-rsa) Key ID: "id"

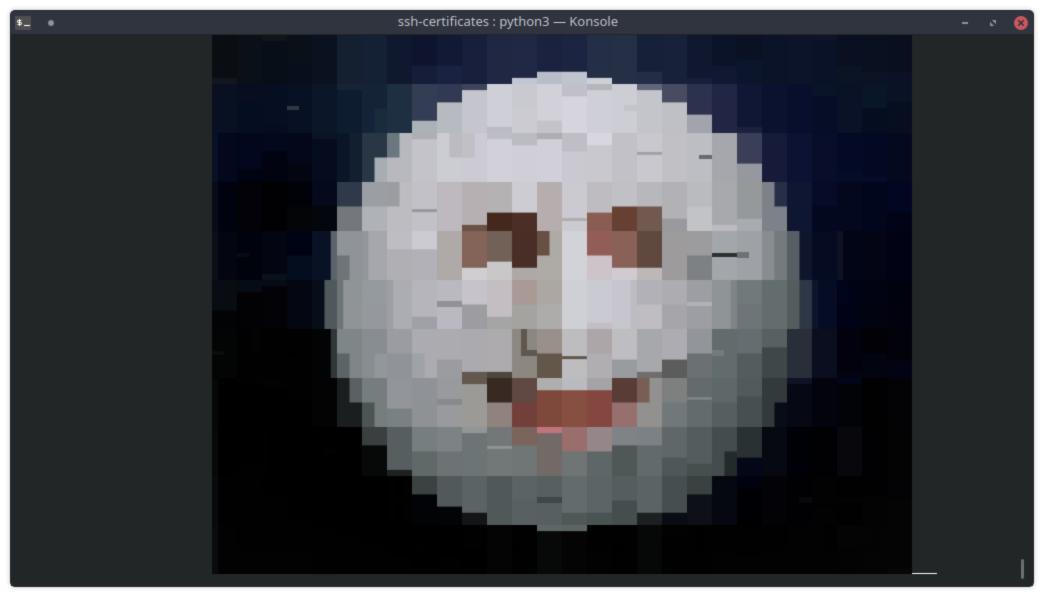
Serial: 0

permit-X11-forwarding
permit-agent-forwarding
permit-port-forwarding
permit-pty
permit-user-rc

man ssh-keygen
man ssh_config
man sshd_config

Interesting features

- * Certificates can expire!
- Force Command
- Source IP Address
- * Permit or deny port/agent forwarding
- * Hosts can have certificates too
- WARNING: IDENTIFICATION HAS CHANGED!



where certificate shine



- 0 (

Onboarding / Offboarding

- * Adding new users is a pain
- * Tools exist:
 - Puppet
 - LDAP
 - Ansible
 - [your favourite tool here]_





Limited Network Access

- * Embedded Systems?
- * Different AWS Accounts?
- * Hosts in different clouds!



As a developer I need to...

I need to SSH to a server and run random SQL queries against the db

Sudo Sessions

pam-ussh by Uber

- * PAM module to grant sudo based on certificate in ssh agent
- * No need to manage user passwords
- * Reduce 2FA fatigue, session aware

Hang on a second...

...am I now just signing certs?

Let's automate that!

python-bless-client - Lyft

- * Uses BLESS, but without a bastion* User laptops sign direct with ca
- * Depends on AWS IAM + KMSAuth

github.com/lyft/python-blessclient

step - Smallstep

* Lightweight Go server to sign certs* Can use OpenIDConnect to auth

github.com/smallstep

vault - Hashicorp

- * Can sign SSH certificates
- * Pluggable authentication backend
- * Need to host Vault

https://www.vaultproject.io/



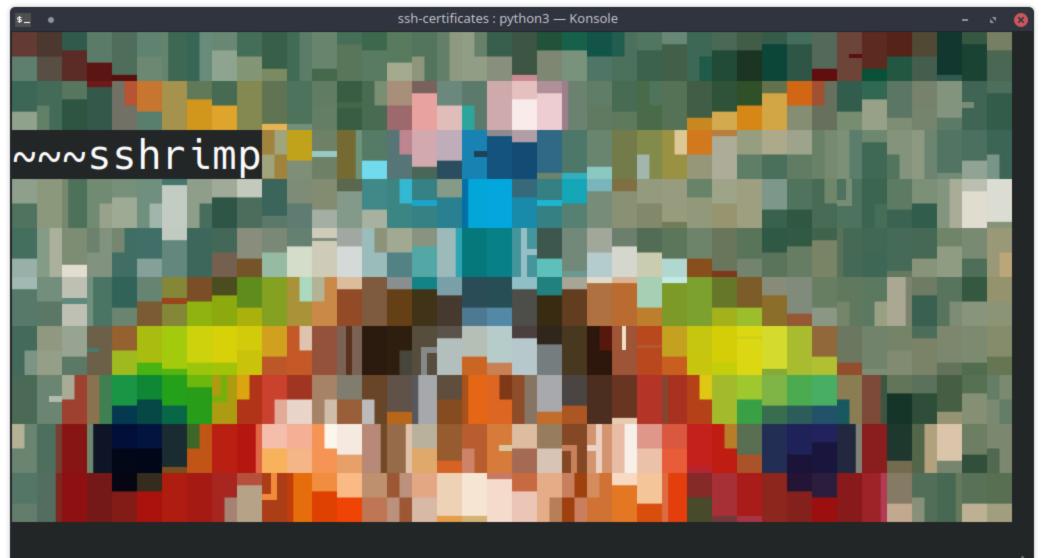
- 2 🤅

Keybase

Chatops Certificate SSH Authority!?!

- * end-to-end encrypted
- * strong verified identity
- * no secrets exchanged

https://keybase.io/blog/keybase-ssh-ca https://github.com/keybase/bot-sshca



sshrimp

demo

|ssh -> ssh-agent (sshrimp-agent)

sshrimp-agent -> OIDC (google)

OIDC -> identity token (JWT)

sshrimp-agent -> sshrimp-lambda in AWS

sshrimp-lambda + JWT + public key = 🎉

```
ssh-certificates: python3 — Konsole
```

"header": { "alg": "RS256", ... },

"email": "jeremy@stott.co.nz",

"signature": "i9ifDU7GkI78pEBcoZ ...

"aud": "430784603061 ...",

"sub": "984203948230948",

"iss": "https://accounts.google.com"

"payload": {

step oauth --oidc --bare | step crypto jwt inspect --insecure

OAuth 2.0 Threat Model and Security Considerations https://tools.ietf.org/html/rfc6819

Proof Key for Code Exchange by OAuth Public Clients https://tools.ietf.org/html/rfc7636

Pesky Users

The user must exist on the host.

- * SSH as root (Facebook?)
- * NSSCache (Google, Lyft)
- * Puppet, but just users?

Issue two certificates to users:

1. user = jeremy

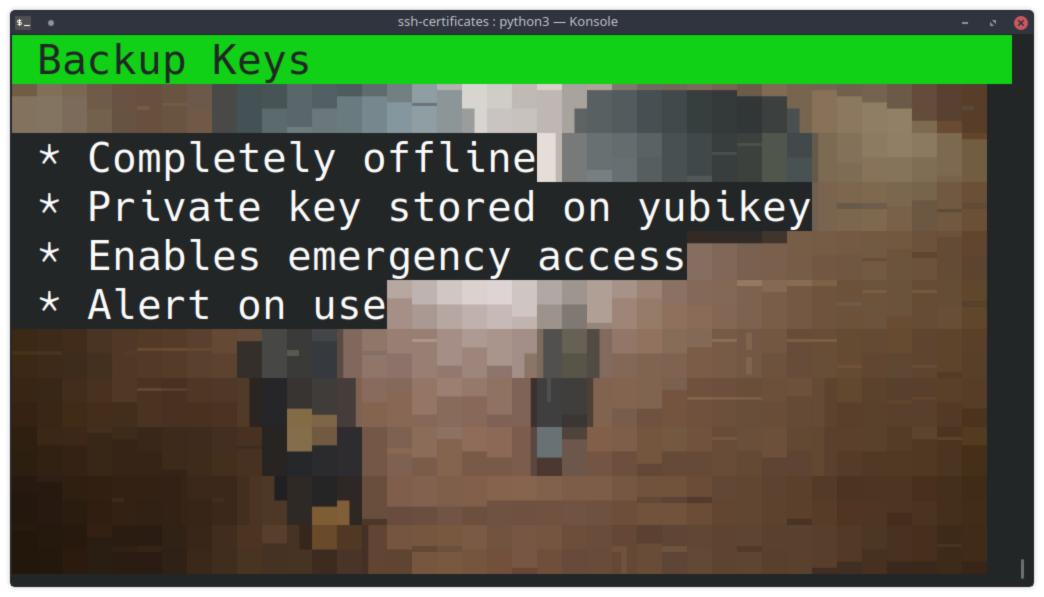
2. user = provision
 force-command = "adduser jeremy"

On the Fly!

```
$ ssh jeremy@host - failed
$ ssh provision@host - accepted
$ ssh jeremy@host - accepted
```

~/.ssh/config Host host

ProxyCommand ssh -T provision@host



SSH Certificates

- * Scalable
- * Strong authentication
- * Increased trust
- * Lightweight (It's just OpenSSH)
- * Improved experience

Bonus...

How did I make these slides?

- * This whole presentation is, inside a SSH certificate!
- * force-command runs a python script

python -c "`echo "G1s5Mm1oZ" | base64 -d | gzip -d`"

Thank you!!



github.com/stoggi/sshrimp

Jeremy Stott jeremy@stott.co.nz @jsstott