

# Updates on SSH with OpenId Connect

EGI Conference 2024

Diana Gudu, Marcus Hardt, Lukas Brocke, Gabriel Zachmann  
Karlsruhe Institute of Technology

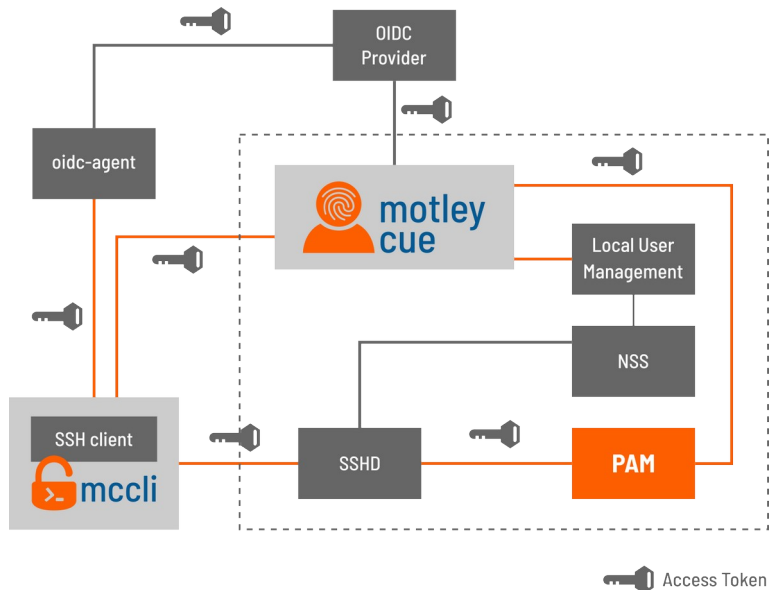
[gudu@kit.edu](mailto:gudu@kit.edu)



# Motivation

- Use benefits of federated identity management with SSH
  - Enable SSH with OpenID Connect
- Address issues related to ssh key management
  - Key approval and distribution
  - Key sharing across devices and teams
  - Keys trusted permanently
  - Passphrases cannot be enforced

# Our solution: recap



## Ecosystem of server & client side tools

- Works with standard SSH software
- Uses OIDC tokens for AuthN & AuthZ
- Server side:
  - PAM module with oidc support: **pam-ssh-oidc\***
  - ID mapping, authZ, dynamic provisioning: **motley-cue**
- Client side:
  - **oidc-agent** for obtaining tokens
  - Enable ssh clients to use tokens: **mccli**, **puTTY**



# Improvements

- Usability
  - Client software installation required
  - Limited commands due to SSH client wrapper
  - What about tools on top of SSH (rsync), or advanced commandlines
- Security
  - TOFU: Trust On First Use for hosts

# Improvements

- Usability
  - Client software installation required
  - Limited commands due to SSH client wrapper
  - What about tools on top of SSH (rsync), or advanced commandlines
- Security
  - TOFU: Trust On First Use for hosts



**Web-shell** in your browser: <https://ssh-oidc-web.vm.fedcloud.eu>

**SSH certificates:** oinit



<https://github.com/dianagudu/webssh-oidc>

# SSH certificates

- SSH certificates are not X.509 certificates
- Data structure that includes:
  - public key, name, expiration, ...
  - **principals**: list of allowed usernames / hostnames
  - **force-command**: allowed command for user
- SSH-CA signs:
  - **user** certificates + **host** certificates
- Clients and hosts trust CA

```
$ ssh-keygen -L -f user-key-cert.pub
user-key-cert.pub:
  Type: ssh-ed25519-cert-v01@openssh.com user certificate
  Public key: ED25519-CERT SHA256:rkSKv...
  Signing CA: ED25519 SHA256:xw9aV... (using ssh-ed25519)
  Key ID: "user@example.com"
  Serial: 0
  Valid: from 2024-09-29T14:30:00 to 2024-09-30T14:30:00
  Principals:
    oinit
  Critical Options:
    force-command oinit-switch diana
  Extensions:
    permit-X11-forwarding
    permit-agent-forwarding
    permit-port-forwarding
    permit-pty
    permit-user-rc
```

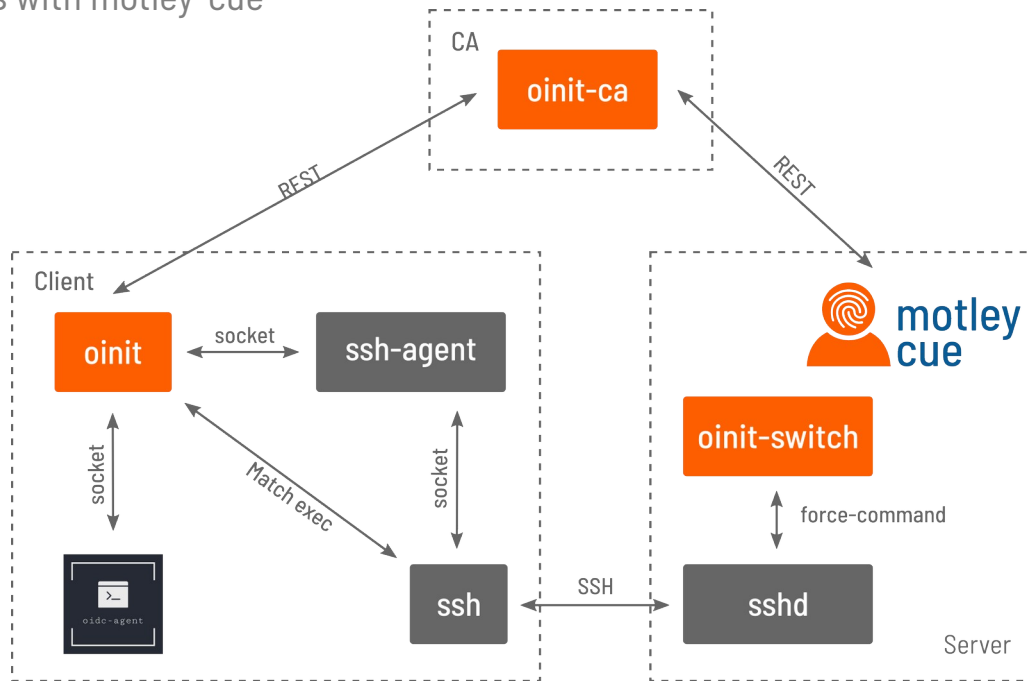


Improve operability: no key approval and distribution, host key management

Improve security: no TOFU, expiration

# oinit

- A new set of tools to integrate SSH certificates with motley-cue



# oinit: CA

- oinit-ca

- Certificate authority that issues user and host certificates
- REST API protected with OIDC token
- Relies on motley-cue for
  - Authorisation
  - Retrieving local username
  - User provisioning

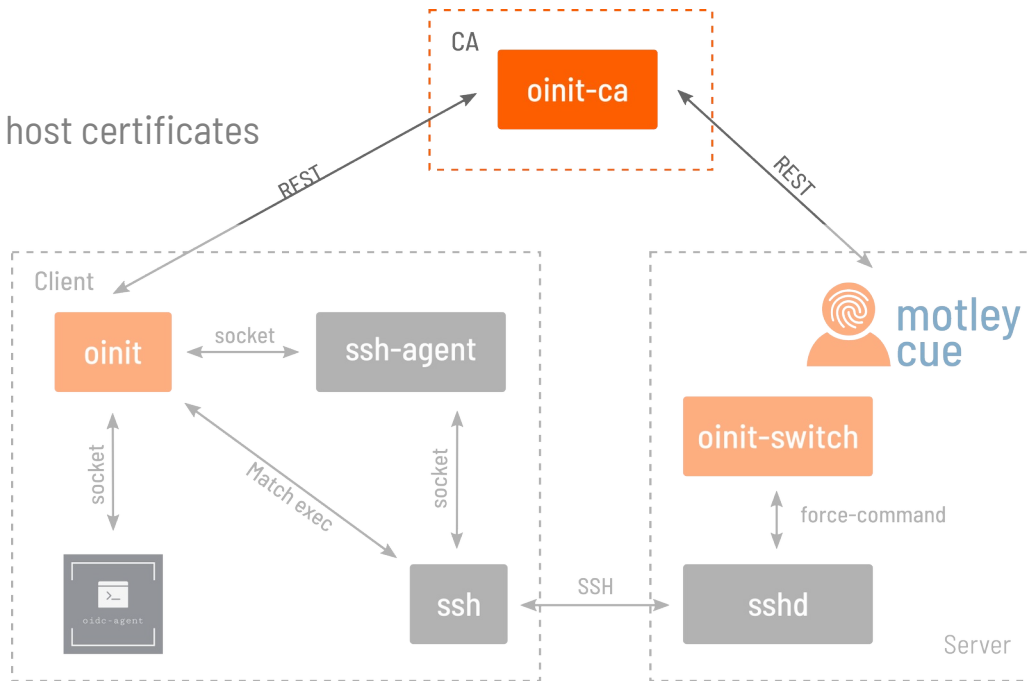
- Certificate issued only for authorised users

Principals:

oinit

Critical Options:

force-command oinit-switch <username>





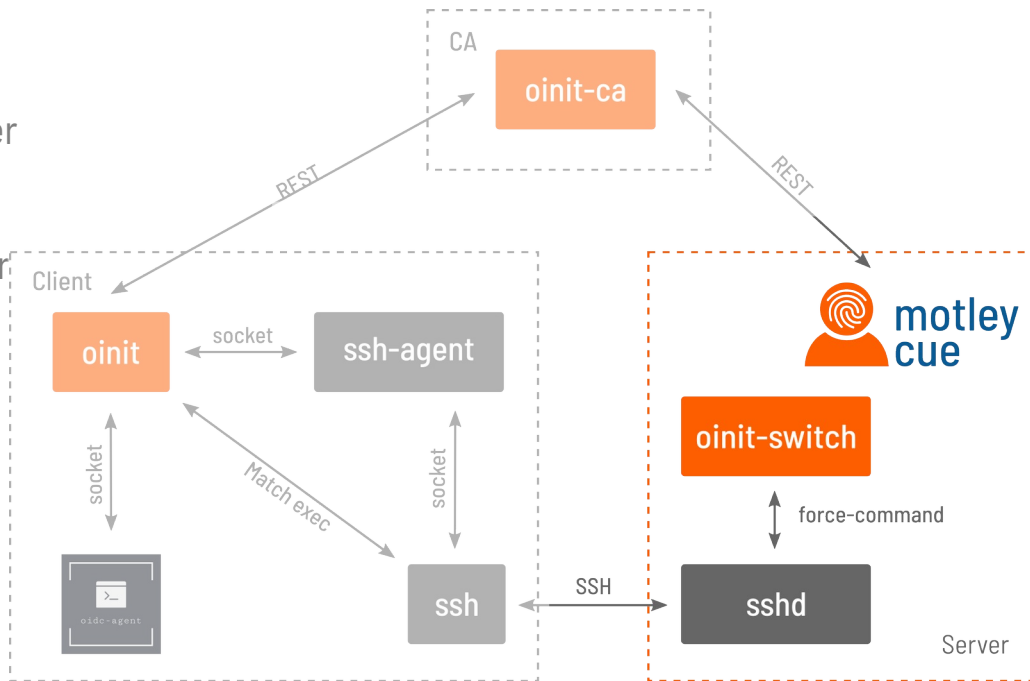
# oinit: server

- oinit-switch

- Transparently switches to the correct user
- Relies on **su**
- Can only be executed by **oinit** service user

- oinit-shell

- custom shell for **oinit** user
- Prevents interactive access
- Restricts allowed commands for user



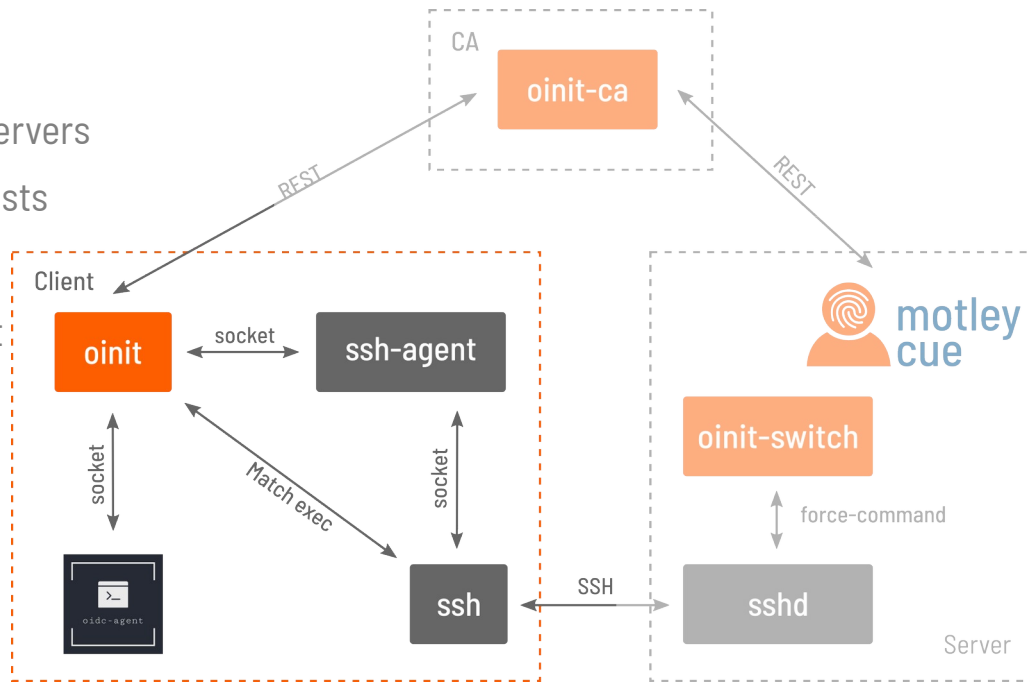
# oinit: client

## oinit

- Configures ssh to use oinit for selected servers
- Configures which CAs to use for which hosts
- Obtains ssh certificates when needed
- Integrates with oidc-agent and ssh-agent

```
Match exec "oinit match %h %p"  
User oinit
```

~/.ssh/config

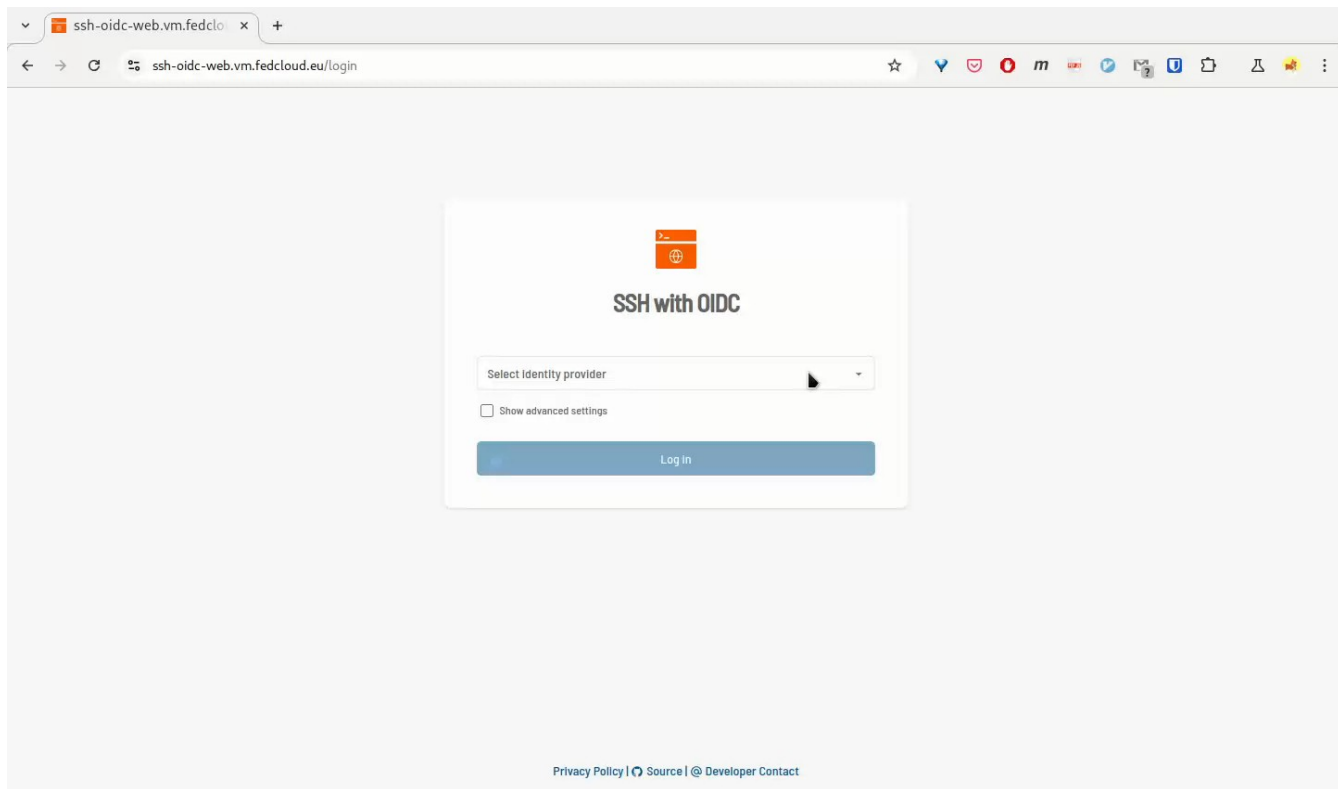




# Demo oinit



# Demo webssh-oidc



The screenshot shows a web browser window with the address bar displaying `ssh-oidc-web.vm.fedcloud.eu/login`. The page content is centered and features a white box with the following elements:

- An orange terminal icon with a white cursor and a plus sign.
- The text **SSH with OIDC**.
- A dropdown menu labeled "Select Identity provider".
- A checkbox labeled "Show advanced settings" which is currently unchecked.
- A blue "Log in" button with a white arrow pointing right.

At the bottom of the page, there is a footer with the text: [Privacy Policy](#) | [Source](#) | [Developer Contact](#).



## Future work

- Improve configuration and interoperability via packaging
- Provide consistent documentation of our ecosystem
- Integration with Account LInking SErvice (ALISE)
- Security audit
  
- Multiple other approaches exist
  - SSH certificates (DEIC), Smart Shell (AWI, SURF), PAM (STFC)
  - Continue working together to make things compatible

## More information

- Documentation:
  - <https://github.com/EOSC-synergy/ssh-oidc>
  - <https://github.com/lbrocke/oinit/blob/main/Documentation/README.md>
- Package repository: <http://repo.data.kit.edu/>
- Demo servers:
  - oinit: [oinit-demo.vm.fedcloud.eu](http://oinit-demo.vm.fedcloud.eu)
  - PAM: <https://ssh-oidc-demo.data.kit.edu/>
  - webssh-oidc: <https://ssh-oidc-web.vm.fedcloud.eu/>
- Contact: [m-contact@lists.kit.edu](mailto:m-contact@lists.kit.edu)



# Backup Slides

# Identity mapping

- Federated ID → local UNIX account
- **motley-cue**
  - Server-side REST interface
  - Authentication via OIDC token
  - Authorisation based on **entitlement** (i.e. VO) + **assurance** + **sub@iss** (user whitelist)
  - Admin interface for security incidents: suspend / resume user
  - Interface to site-local identity management systems
- **Dynamic provisioning** supported if mapping does not exist
  - Pooled-accounts, “Friendly” username
  - VOs mapped to local groups
  - Multiple provisioning backends: local, LDAP, ticket system



<https://motley-cue.readthedocs.io/>





# Token-based authentication

- **pam-ssh-oidc**

- Standard UNIX interface, simple and well-understood
- Can be chained with other PAM modules (e.g. MFA)
- Prompts for Access Token instead of Password
- Validates Access Token, user & authorisation with motley-cue

```
$ ssh testuser@ssh-oidc-demo.data.kit.edu  
(testuser@ssh-oidc-demo.data.kit.edu) Access Token:
```

# User experience

- Obtain Access Token
  - **oidc-agent**: secure, largely non-interactive
- Get local username (and optionally provision user)
  - HTTP calls to motley-cue API with token
- Automation of client tasks: **mccli**
  - oidc-agent integration and forwarding
  - Non-interactive: pass token to ssh when prompted
  - OTP: via motley-cue for long tokens > 1024



<https://indigo-dc.gitbook.io/oidc-agent>



<https://mccli.readthedocs.io/>

```
$ mccli ssh ssh-oidc-demo.data.kit.edu --oidc egi
testuser@ssh-oidc-demo:~$
```