

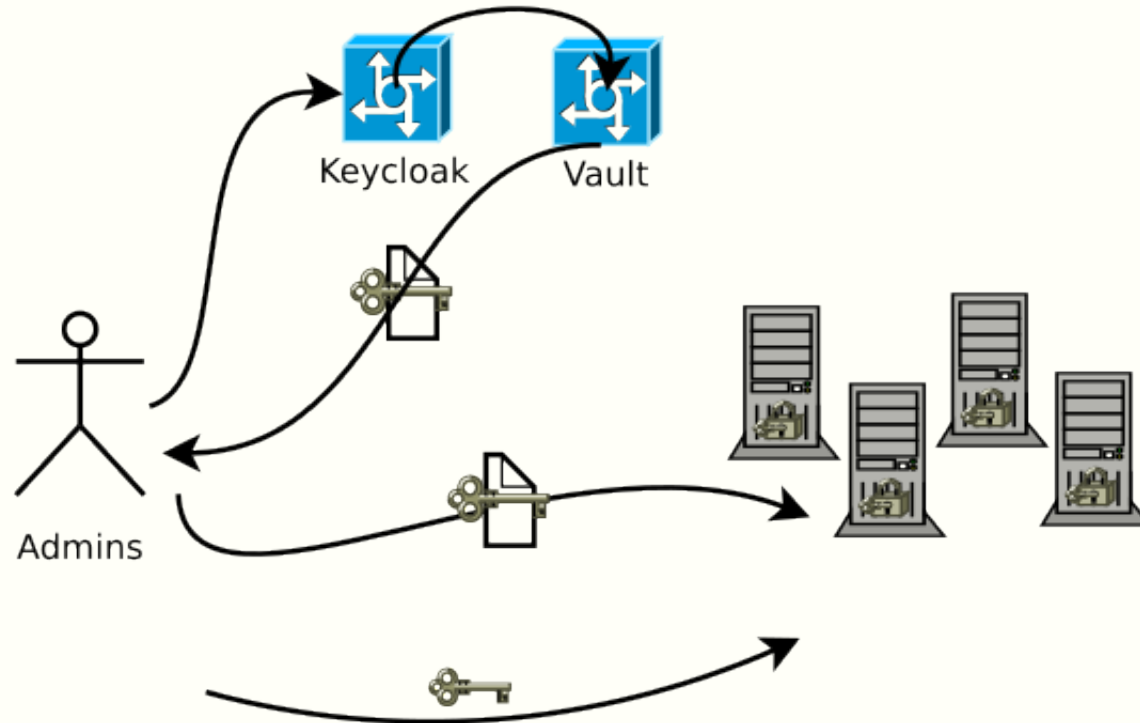
Login mit signierten ssh- Schlüsseln

Christopher J. Ruwe <cjr@cyberiada.de>
Cyberiada GmbH

Problem

- ohne großen Aufwand sollen Personen zum Login auf eine (große) Menge an Hosts berechtigt werden
- Rechte sollen einfach und schnell entzogen werden können
- es soll keine aufwendige Authn/Authz-Infra (AD/krb) gepflegt werden

Lösungs-Angebot



Lösungs-Angebot

Type: ssh-rsa-cert-v01@openssh.com user certificate
Public key: RSA-CERT SHA256:tJYvjQig65sxMxlcoH9MJyboWu2Gru092ORNfh1XhNs
Signing CA: RSA SHA256:tZIipsTz7DJE6kGbESP5d+sNvgryqK3fWPiIrDvzhFk (using rsa-sha2-256)
Key ID: "vault-oidc-belphegor@he.ll<long_number>"
Serial: 16626502850102714285

Valid: from 2020-11-14T17:41:51 to 2020-12-16T17:42:21

Principals:

hal9000

Critical Options: (none)

Extensions:

permit-X11-forwarding

permit-agent-forwarding

permit-port-forwarding

permit-pty

permit-user-rc

wie hilft das?

- Schlüssel-Signatur nur gegeben Token von zentralem authn/authz Mechanismus
- Gültigkeit kurz
- Schlüssel ungültig heißt Login-Rechte weg

wie läuft das?

- keine persönlichen Nutzer mehr
- nur noch Funktionsnutzer mit entsprechenden Rechten
- Zuordnung über Audit-Logs

Mechanismus

- Nutzer-Datenbank entweder direkt oder *federated* in Keycloak
- dort Pflege von Gruppen-Zuordnung
- Übermittlung im OIDC-Token

Identität und Rechte-Verwaltung

The screenshot displays the Keycloak administration interface. At the top left is the Keycloak logo. The breadcrumb navigation shows 'Users > belphegor'. The user's name 'Belphegor' is displayed with a trash icon. Below this are tabs for 'Details', 'Attributes', 'Credentials', 'Role Mappings', 'Groups', 'Consents', and 'Sessions'. The 'Groups' tab is active. It contains two panels: 'Group Membership' and 'Available Groups'. The 'Group Membership' panel has a search bar, a 'View all groups' button, and a 'Leave' button. It lists two groups: '/host-admins' and '/vault-admins'. The 'Available Groups' panel also has a search bar, a 'View all groups' button, and a 'Join' button. It lists two groups: 'host-admins' and 'vault-admins'. On the left side, there is a dark sidebar with a 'Configure' section (Realm Settings, Clients, Client Scopes, Roles, Identity Providers, User Federation, Authentication) and a 'Manage' section (Groups, Users, Sessions, Events, Import, Export). The 'Users' option is highlighted. In the top right corner, the user 'Admin' is logged in.

Token-Mapping

Clients > vault-client

Vault-client

- Settings
- Credentials
- Roles
- Client Scopes**
- Mappers
- Scope
- Revocation
- Sessions
- Offline Access
- Clustering
- Installation

Setup **Evaluate**

Scope Parameter openid

Client Scopes Available Optional Client Scopes

- address
- microprofile-jwt
- offline_access
- phone

Add selected >

Selected Optional Client Scopes

<< Remove selected

Effective Client Scopes

- email
- groups
- profile
- roles
- web-origins

User belphegor

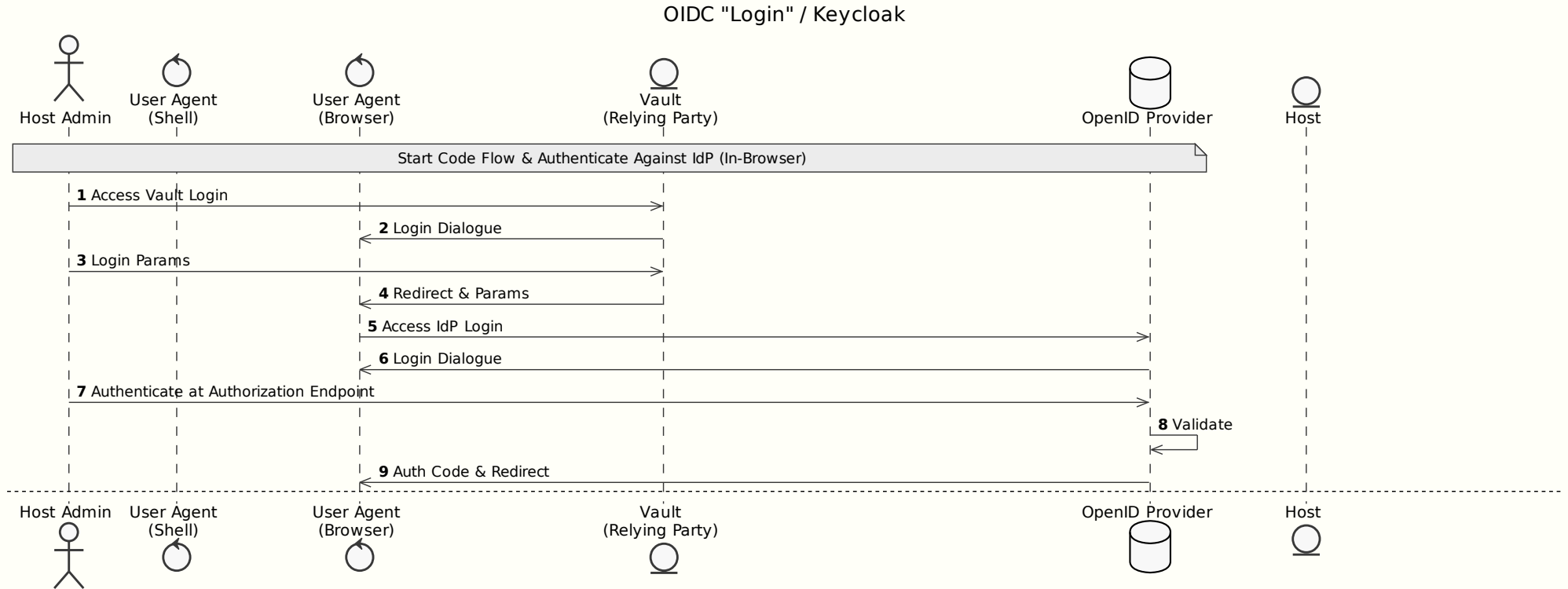
Evaluate

Effective Protocol Mappers Effective Role Scope Mappings **Generated Access Token**

```
},
"scope": "openid profile email groups",
"email_verified": true,
"name": "Baal Peor",
"groups": [
  "/host-admins",
  "/vault-admins"
],
"preferred_username": "belphegor",
"given_name": "Baal",
"family_name": "Peor",
"email": "belphegor@he.ii"
}
```

Verfahren

Sign SSH Keys



Gruppen-Zuordnung

The screenshot shows a web application interface for group management. At the top, there is a navigation bar with a search icon, tabs for 'Secrets', 'Access', 'Policies', and 'Tools', and a 'Status' indicator. The left sidebar contains a menu with 'ACCESS' at the top, followed by 'Auth Methods', 'Entities', 'Groups' (highlighted with a blue bar), and 'Leases'. The main content area shows a breadcrumb for 'Groups' and the title 'host-admins'. Below the title are tabs for 'Details', 'Aliases', 'Policies' (active), 'Members', 'Parent groups', and 'Metadata'. A large light gray box contains an 'Edit group >' button. Below this is a table with one row for 'host-admins' and a three-dot menu icon to its right.

Secrets Access Policies Tools Status

ACCESS

Auth Methods

Entities

Groups

Leases

< Groups

host-admins

Details Aliases Policies Members Parent groups Metadata

Edit group >

<u>host-admins</u>	...
--------------------	-----

Rechte-Vergabe per Policy

The screenshot shows a web interface with a dark top navigation bar. The navigation bar contains a search icon, the text 'Secrets Access Policies Tools', a 'Status' indicator with a dropdown arrow, and two user profile icons with dropdown arrows. Below the navigation bar, the main content area is divided into a left sidebar and a main panel. The sidebar has a 'POLICIES' header and a sub-section 'ACL Policies' which is highlighted with a blue vertical bar. The main panel shows a breadcrumb '< ACL policies', a title 'host-admins', and two buttons: 'Download policy >' and 'Edit policy >'. Below these is a code editor showing a policy in HCL format:

```
1 path "fraosug-demo/sign/host-admins-ssh" {
2   capabilities = [
3     "update"
4   ]
5 }
6
```

Transport per OIDC

Secrets **Access** Policies Tools Status > > >

ACCESS

Auth Methods

Entities

Groups

Leases

[Entity aliases](#)

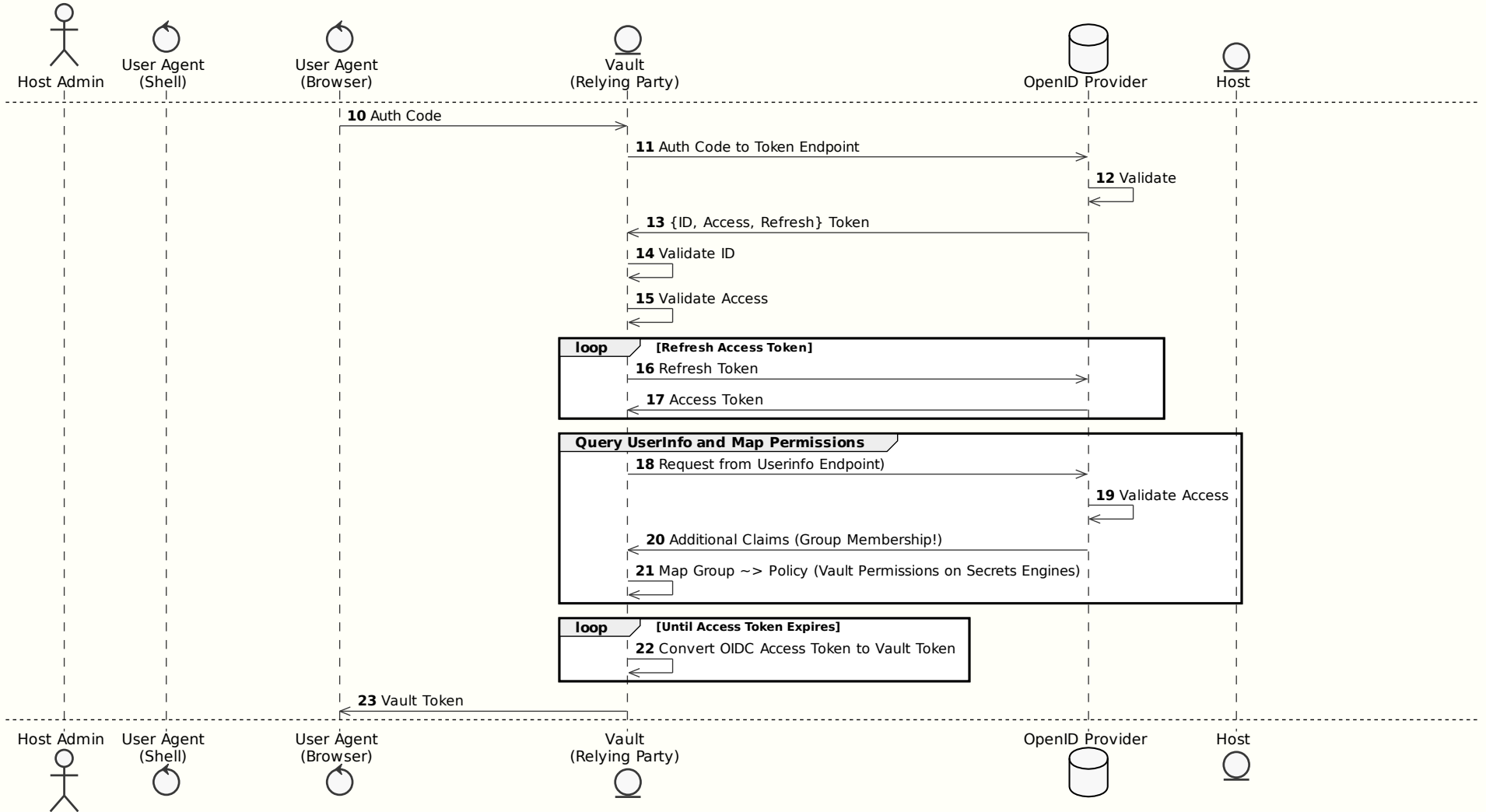
belphegor@he.ll

[Details](#) [Metadata](#)

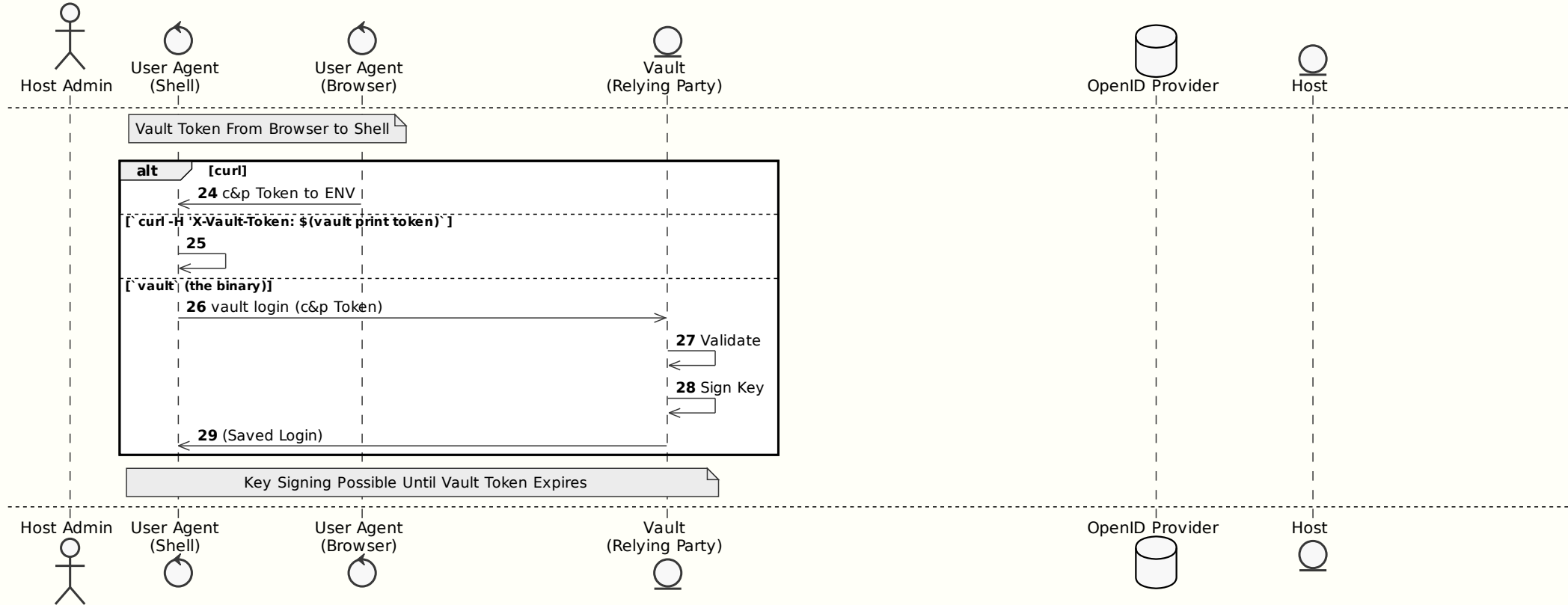
[Edit entity alias](#) >

Name	belphegor@he.ll
ID	0c5bd4c1-9d3e-d3ca-7f21-f3e841966881
Entity ID	<u>d9c5a79a-7673-b92c-413c-93b1c082080b</u>
Mount	auth/oidc/ oidc auth_oidc_48ae3a32
Created	Nov 14, 2020 at 4:56 pm
Last Updated	Nov 14, 2020 at 4:56 pm

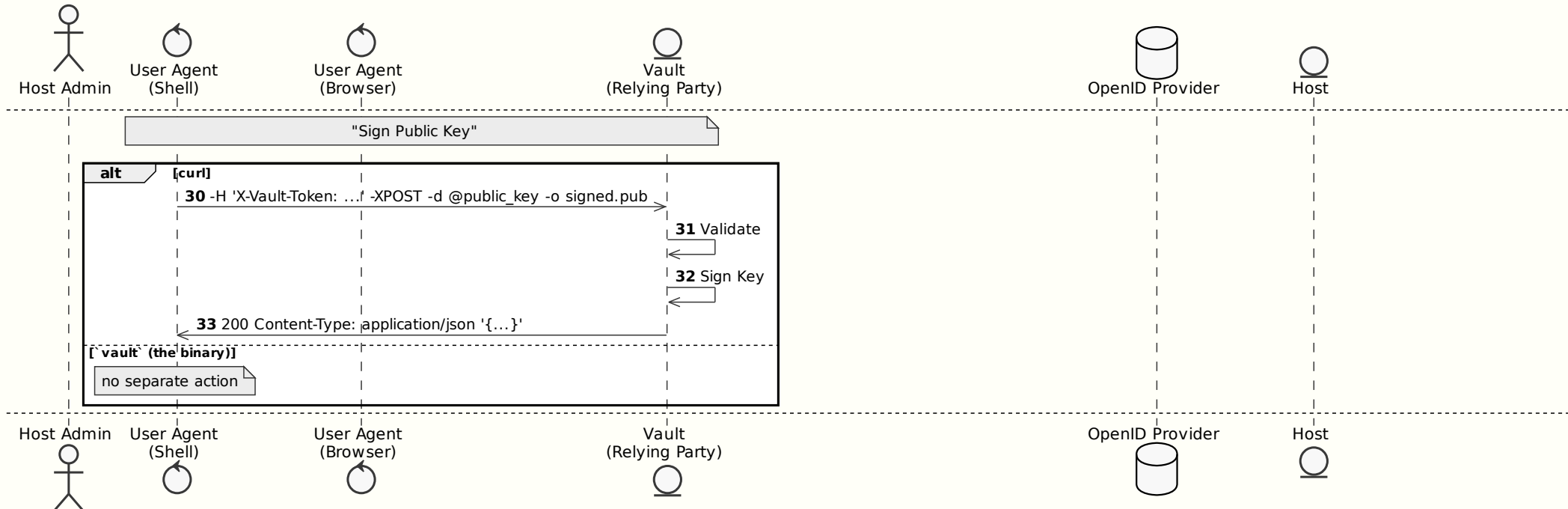
Transport Authentication & UserInfo to Vault



Use Vault Token



How to Sign Keys



Rechte-Übersetzung per Rolle

[fraosug-demo](#) < [host-admins-ssh](#)

SSH role host-admins-ssh

	Sign Keys > Delete role v Edit role >
Role Name	host-admins-ssh
Key type	ca
Allow user certificates	<input checked="" type="checkbox"/> Yes
Allow host certificates	<input checked="" type="checkbox"/> Yes
Default Username	hal9000
Allowed users	hal9000
Default critical options	{}
Allowed extensions	permit-pty, permit-agent-forwarding, permit-user-rc, permit-port-forwarding, permit-X11-forwarding
Default extensions	<pre>{ "permit-agent-forwarding": "", "permit-pty": "", "permit-user-rc": "" }</pre>

Signatur eines Schlüssels

```
{
curl \
  -X PUT \
  -H "X-Vault-Request: true" \
  -H "X-Vault-Token: $(vault print token)" \
  -d@- \
  http://192.168.99.67:8200/v1/fraosug-demo/sign/host-admins-ssh \
<<EOT
{
  "cert_type":"user",
  "extensions": {
    "permit-X11-forwarding": "",
    "permit-agent-forwarding": "",
    "permit-port-forwarding": "",
    "permit-pty": "",
    "permit-user-rc": ""
  },
  "public_key": "$(cat /home/cjr/media/src/50-edu/50-talks/201117-cjr--fraosug-ssh/doc/fraosug.pub)",
  "valid_principals": "hal9000"
}
EOT
} | jq \
  -r '.data.signed_key' \
| ssh-keygen -f - -L
```

Login

```
ssh \
  -i ../fraosug \
  -oCertificateFile=../fraosug-signed.pub \
  hal9000@192.168.99.68
```

```
vault ssh \
  -mode=ca \
  -mount-point=fraosug-demo \
  -role=host-admins-ssh \
  hal9000@192.168.99.68
```

Audit-Trail (I)

Nov 14 17:12:37 moon sshd[5716]: Postponed publickey for hal9000 from 192.168.99.1 port 40154 ssh2 [preauth]

Nov 14 17:12:37 moon sshd[5716]: Accepted publickey for hal9000 from 192.168.99.1 port 40154 ssh2: RSA-CERT

SHA256: TGvFsbj58aEX1hP3RMSUT/7t4y5lorasgfr9xhLYZwg ID vault-oidc-belphegor@he.ll-4c6bc5b1b8f9f1a117d613f744c4944ffeede32e65a2b6ac81fafdc612d86708 (serial 6705468412955462474) CA RSA

SHA256: rW3eVJ1WXeCc/rou0g3MXIJXHlOYcJdf10/nYDWX250

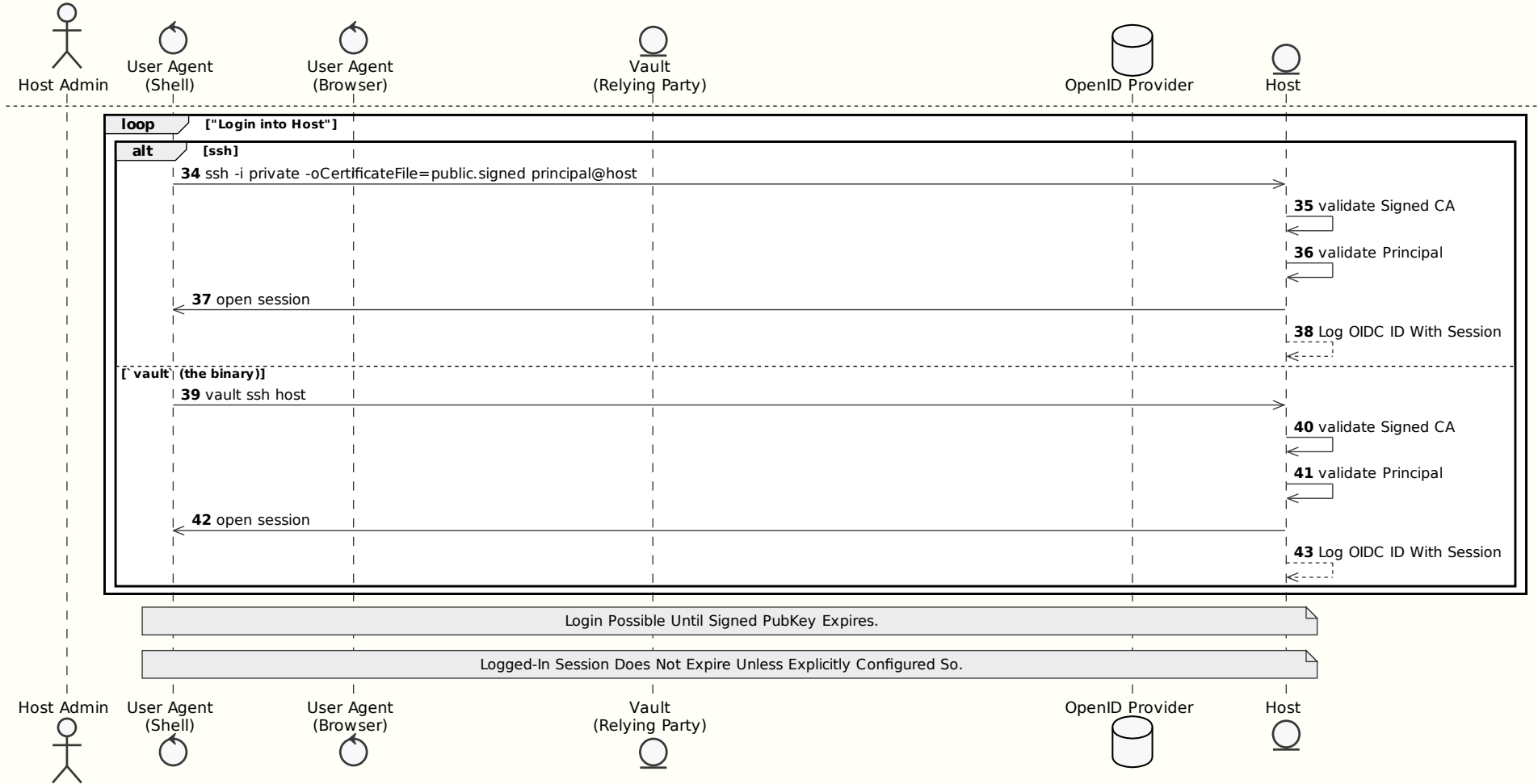
Nov 14 17:12:37 moon sshd[5716]: pam_unix(sshd:session): session opened for user hal9000 by (uid=0)

Nov 14 17:12:37 moon systemd-logind[706]: New session 31 of user hal9000.

Audit-Trail (2)

```
type=USER_LOGIN msg=audit(1605373958.051:430): pid=5716 uid=0 auid=1000 ses=31
msg='op=login id=1000 exe="/usr/sbin/sshd" hostname=192.168.99.1
addr=192.168.99.1 terminal=/dev/pts/1 res=success'
type=USER_ACCT msg=audit(1605374084.981:439): pid=5827 uid=1000 auid=1000 ses=31
msg='op=PAM:accounting grantors=pam_permit acct="hal9000" exe="/usr/bin/sudo"
hostname=? addr=? terminal=/dev/pts/1 res=success'
type=USER_CMD msg=audit(1605374084.981:440): pid=5827 uid=1000 auid=1000 ses=31
msg='cwd="/home/hal9000" cmd="ls" terminal=pts/1 res=success'
type=CRED_REFR msg=audit(1605374084.981:441): pid=5827 uid=0 auid=1000 ses=31
msg='op=PAM:setcred grantors=pam_permit,pam_cap acct="root" exe="/usr/bin/sudo"
hostname=? addr=? terminal=/dev/pts/1 res=success'
```

Logging in to Hosts



Fallstricke

- es handelt sich beim signierten Schlüssel konzeptionell um ein Zertifikat, **kein Token**
- auch wenn das Zertifikat per `-i . . .`` übergeben werden kann, ist es kein neues Secret

Exkurs: terraform

- API-Möhre
- *recte*: Erzeugung und Konfiguration von Ressourcen über eine API
- ursprünglich Cloud-Ressourcen, mittlerweile alles mögliche inkl. Pizza-Bestellung

Exkurs: terraform

```
terraform {
  required_providers {
    keycloak = {
      source = "mrparkers/keycloak"
      version = "2.0.0"
    }

    vault = {
      source = "hashicorp/vault"
      version = "2.15.0"
    }
  }
  backend "pg" {}
}
```

```
provider "keycloak" {
  client_id = "admin-cli"
  username = "terraform"
  password = "..."
  url = var.endpoint-keycloak
}

provider "vault" {
  address = var.endpoint-vault
  skip_tls_verify = true
  token = "...."
}
```

Exkurs: terraform

```
{  
  "version": 3,  
  "serial": 1,  
  "lineage": "4fc1a10d-9a41-e161-353a-06418f6d2f59",  
  "backend": {  
    "type": "pg",  
    "config": {  
      "conn_str": "postgres://terraform:imustshootyouafteryouhavereadthis-<...>@192.168.99.66/terraform",  
      "schema_name": null,  
      "skip_schema_creation": null  
    },  
    "hash": 241288904  
  },  
  "modules": [ <...> ]  
}
```