



# Managing user identity on Macs



**Sean Rabbitt**

Sr Consulting Engineer,  
Identity and Access Mgmt

PRESENTING TO  
**2023 MACADMIN'S  
CONFERENCE**



# Agenda

## 1 | Background and history of macOS

I promise not to bore you with stories of how I used to work at Data General and DG/UX

## 2 | Local User Accounts

How to deal with them, command line fun times, and why we're stuck with them forever. (Spoiler: FileVault)

## 3 | On-Premises and Cloud Directories

Where Sean goes on a rant about binding, the alternatives, and cloud identity provider management

## 4 | The Future: Platform Single Sign-On

With a whole bunch of speculation because after 4 years, we barely have normal Single Sign-On





# A short history lesson



# History





# History

# macOS is UNIX





# macOS is **UNIX**



## Local Accounts and Groups

Short Name  
Real Name  
UID  
Primary Group  
Home Directory



## Hierarchical File Structure

File Owner  
Group Owner  
Read / Write / Execute  
Other Apple Specific Magic

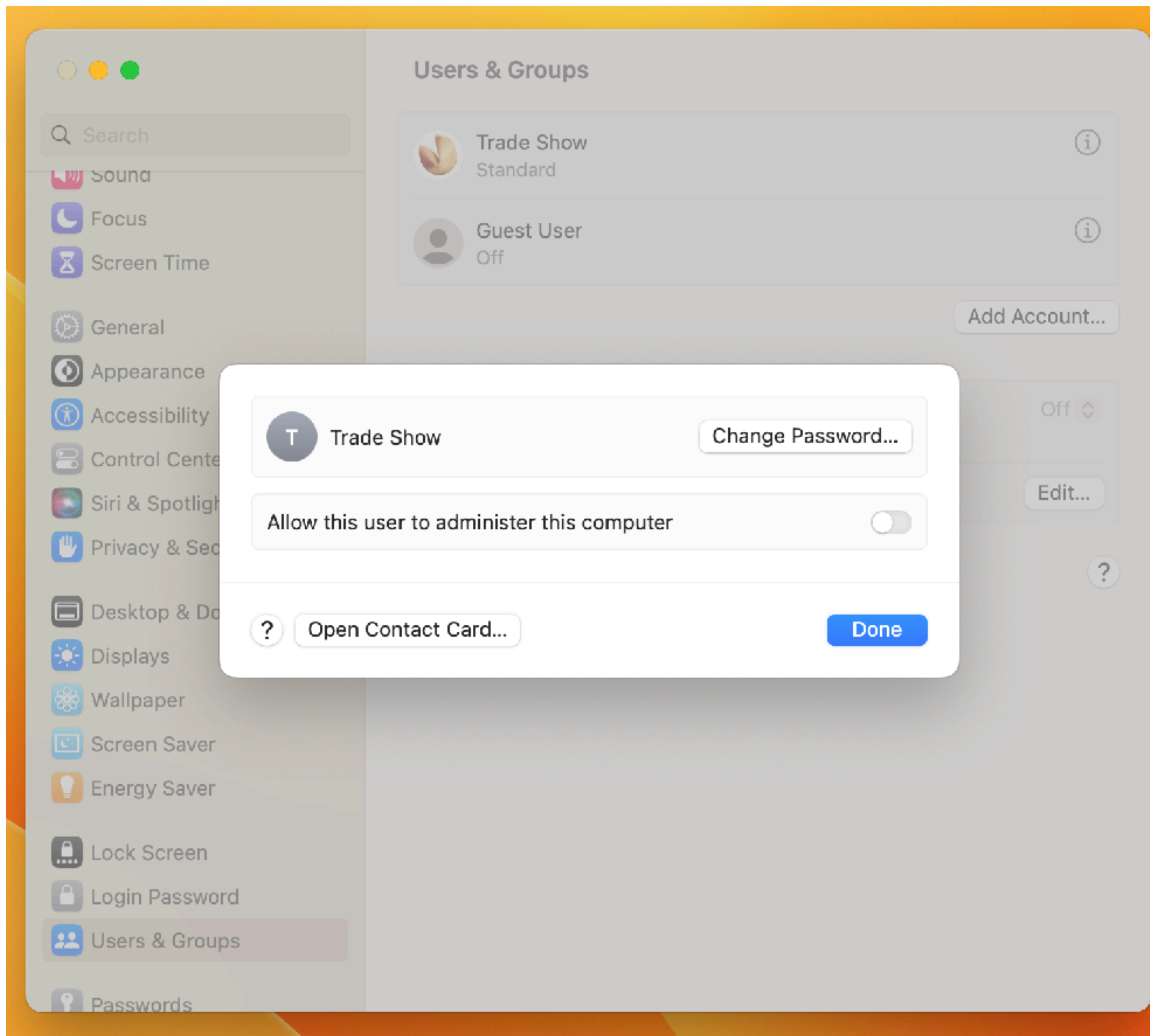


## Basic Privilege Access Management (PAM)

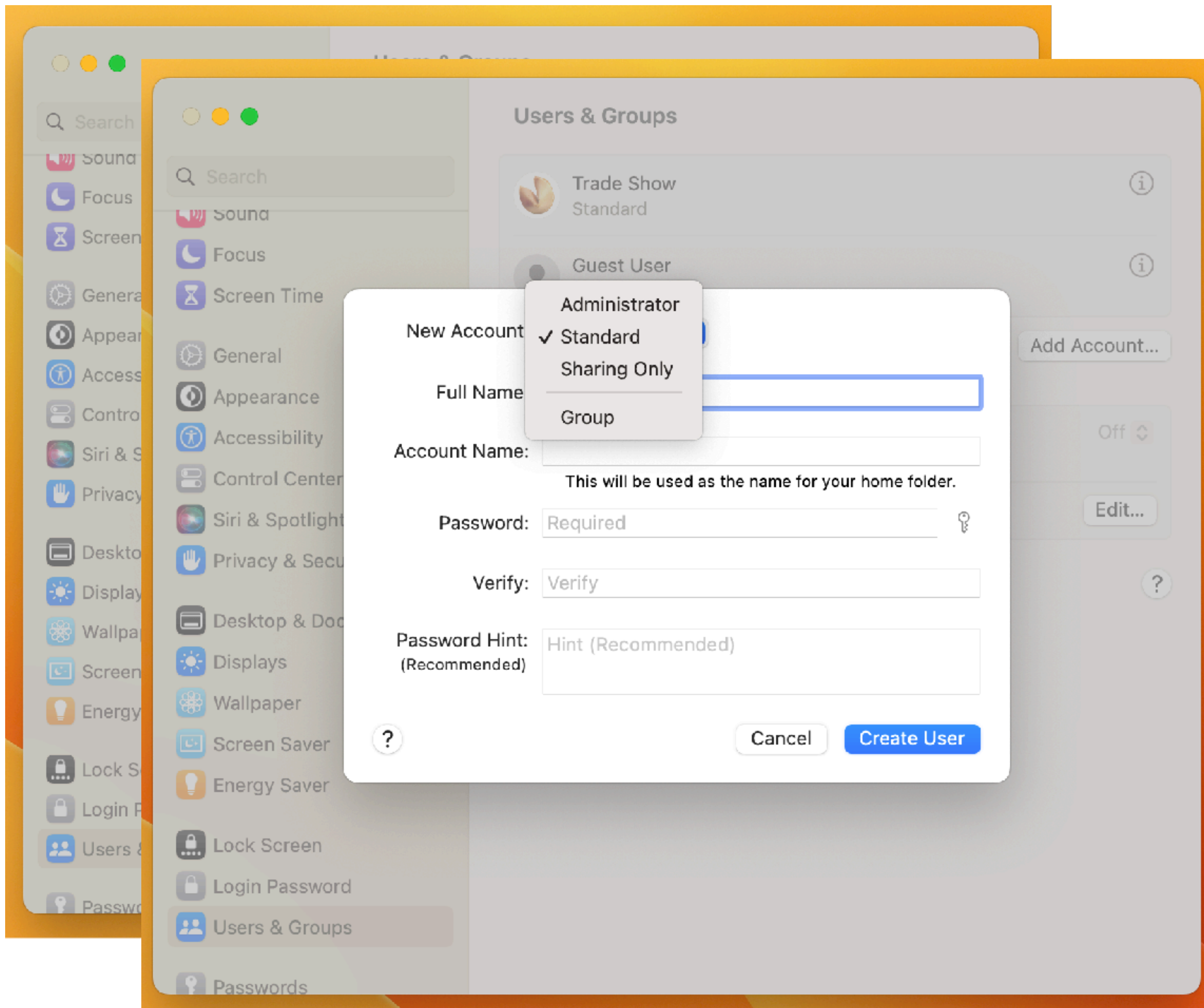
Administrator User  
Standard User  
Guest User  
Sharing Only User



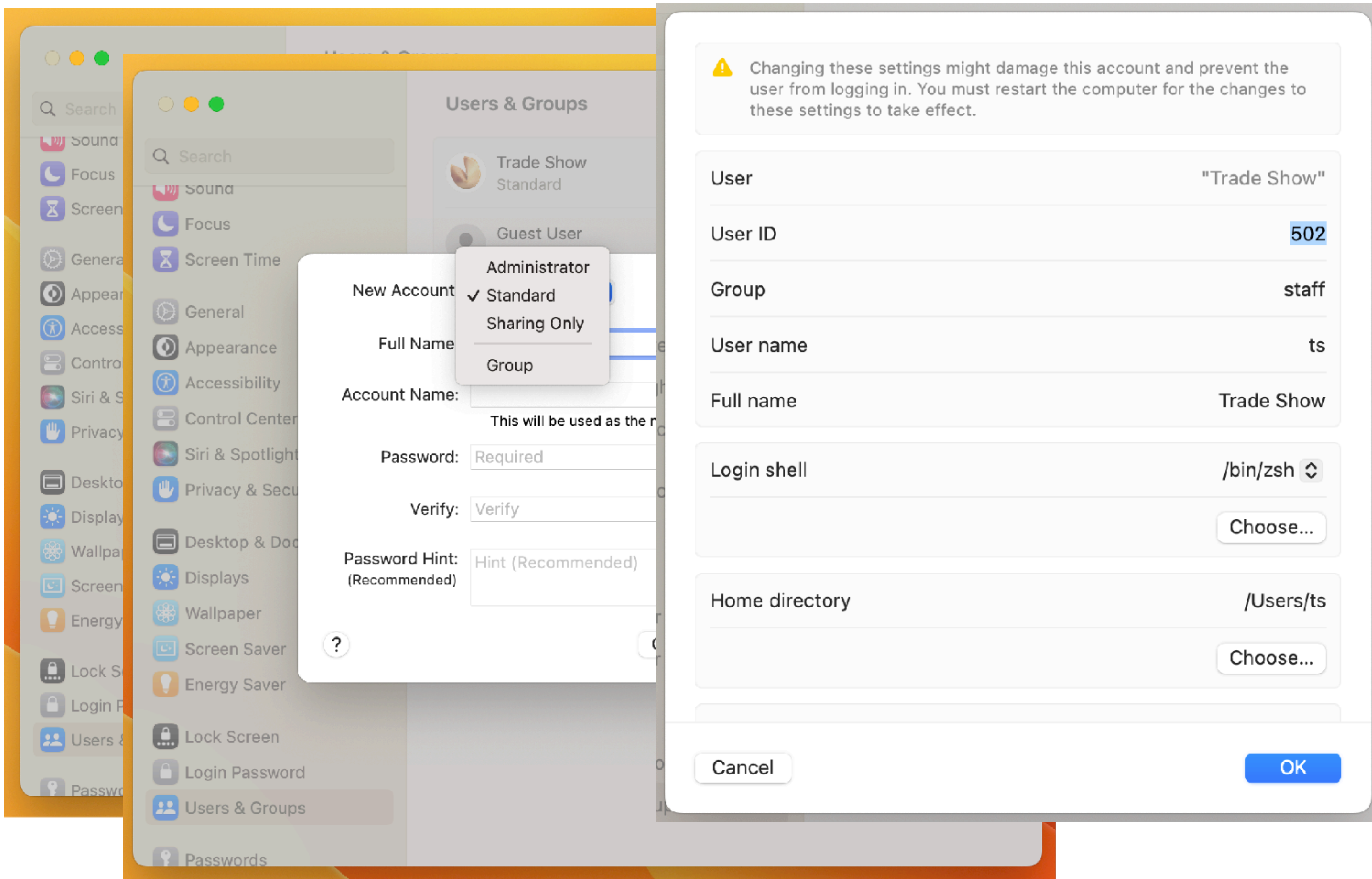
# Local User Accounts



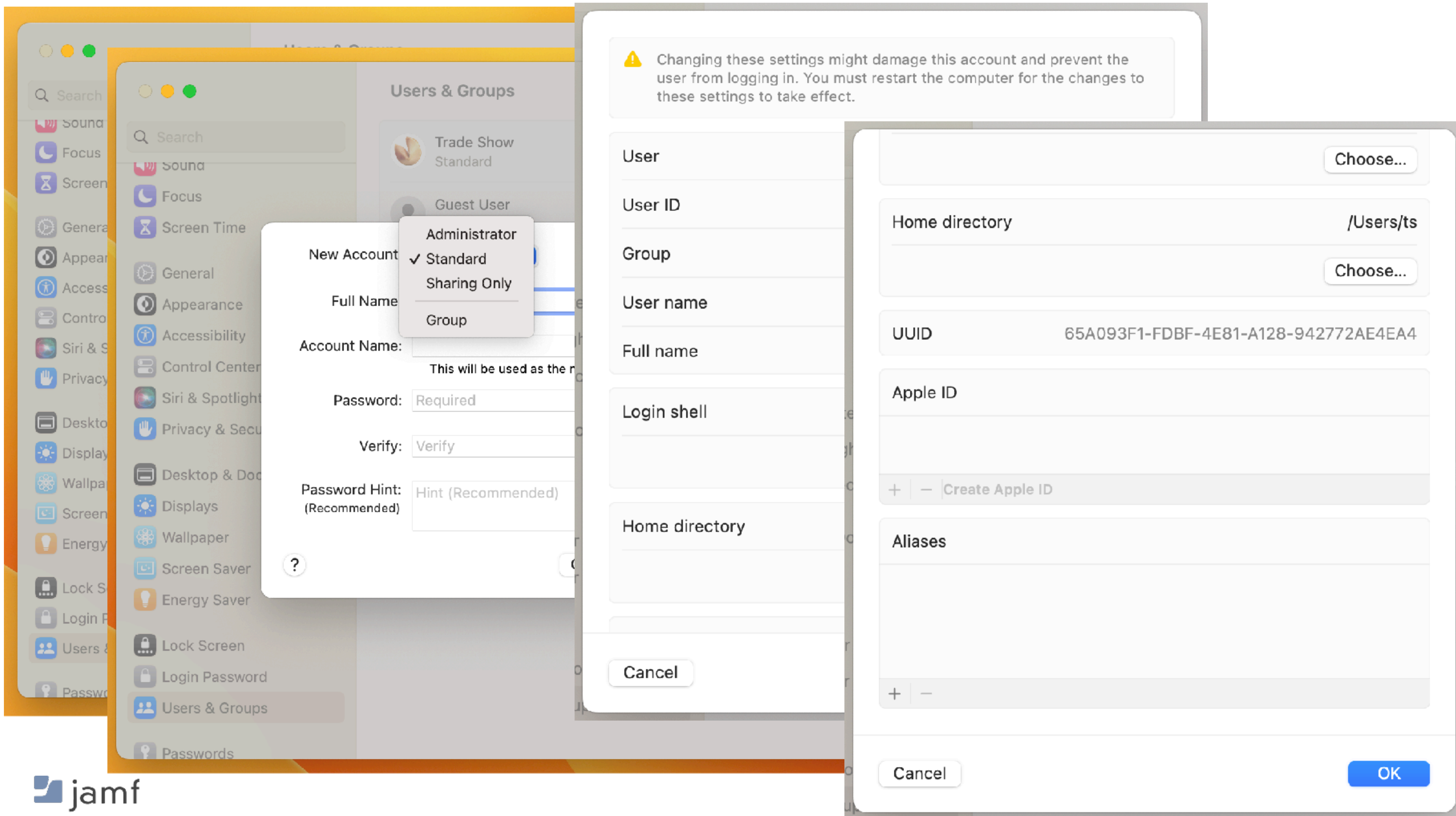




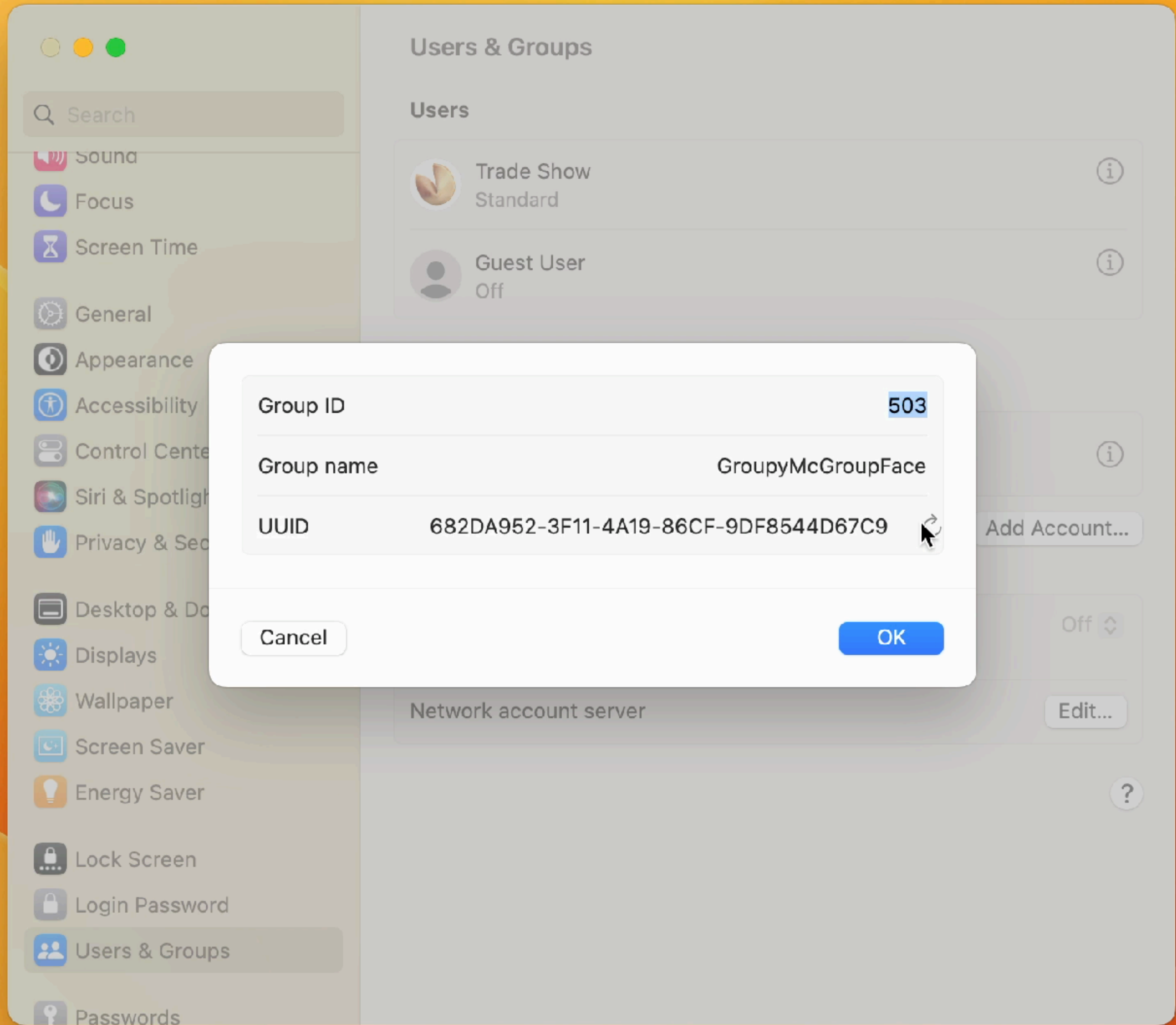




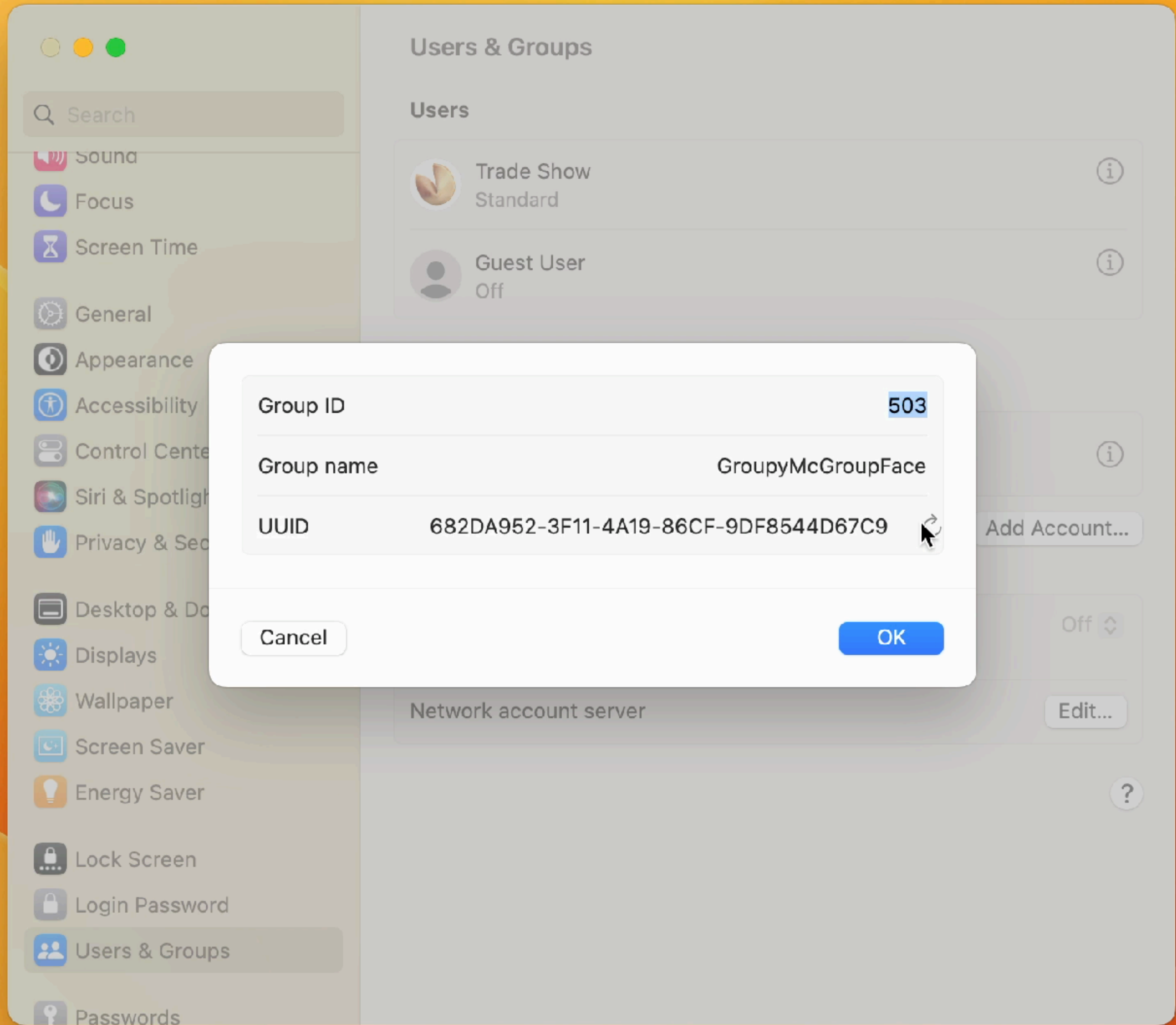















 Changing these settings might damage this account and prevent the user from logging in. You must restart the computer for the changes to these settings to take effect.

User "Trade Show"

User ID 502

Group staff

User name ts

Full name Trade Show

Login shell /bin/zsh 

Choose...

Home directory /Users/ts

Choose...

Cancel

OK

Choose...

Home directory /Users/ts

Choose...

UUID 65A093F1-FDBF-4E81-A128-942772AE4EA4

Apple ID

+ - Create Apple ID

Aliases

+ -

Cancel

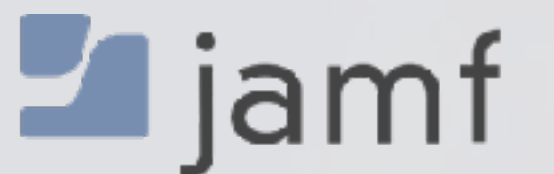
OK



“Hey, sometimes I’m lazy and I like to use a GUI. Then I realize that I need to get back to work.”

Sean Rabbitt

SENIOR CONSULTING ENGINEER, IDENTITY AND ACCESS MANAGEMENT, JAMF



Joke gratuitously stolen from Tim Knox



# To Thine Own Self Be True, or who am i, really?

```
whoami
```

```
echo $USER
```

```
loggedInUser=$(stat -f %Su /dev/console)  
echo "$loggedInUser"
```

```
loggedInUser=$(scutil <<< "show State:/Users/ConsoleUser" \  
| awk '/Name :/ && ! /loginwindow/ { print $3 }' )  
echo "$loggedInUser"
```

# dsc1



# dsc1

```
dsc1 . read /Users/$user
```

# dsc1

dsc1 . read /Users/\$user

```
ts — -zsh — 181x52

dsAttrTypeNative:writers_hint: ts
dsAttrTypeNative:writers_jpegphoto: ts
dsAttrTypeNative:writers_passwd: ts
dsAttrTypeNative:writers_picture: ts
dsAttrTypeNative:writers_unlockOptions: ts
dsAttrTypeNative:writers_UserCertificate: ts
dsAttrTypeNative:accountPolicyData:
  <?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
  <plist version="1.0">
  <dict>
    <key>creationTime</key>
    <real>1687821212.484699</real>
    <key>failedLoginCount</key>
    <integer>0</integer>
    <key>failedLoginTimestamp</key>
    <integer>0</integer>
    <key>passwordLastSetTime</key>
    <real>1687821212.507021</real>
  </dict>
</plist>

dsAttrTypeNative:AvatarRepresentation:
dsAttrTypeNative:record_daemon_version: 8780000
dsAttrTypeNative:unlockOptions: 0
AppleMetaNodeLocation: /Local/Default
AuthenticationAuthority: ;SecureToken; ;ShadowHash;HASHLIST:<SALTED-SHA512-PBKDF2,SRP-RFC5054-4096-SHA512-PBKDF2> ;Kerberosv5;;ts@LKDC:SHA1.8DCD22811DA43DBA95A290C16E6FAF928CE94D09;
LKDC:SHA1.8DCD22811DA43DBA95A290C16E6FAF928CE94D09;
GeneratedUID: 65A093F1-FDBF-4E81-A128-942772AE4EA4
NetworkSignIn:
  2023-06-26 23:13:32 +0000
NetworkUser: ts@jamfse.io
NFSHomeDirectory: /Users/ts
OIDCProvider: Azure
Password: *****
Picture:
  /Library/User Pictures/Fun/Fortune Cookie.heic
PrimaryGroupID: 20
RealName:
  Trade Show
RecordName: ts
RecordType: dsRecTypeStandard:Users
UniqueID: 502
UserShell: /bin/zsh
ts@H2WGW2C9Q6NV ~ %
```



# dsc1

```
dsc1 . read /Users/$user
```

```
ts — -zsh — 181x52

dsAttrTypeNative:writers_hint: ts
dsAttrTypeNative:writers_jpegphoto: ts
dsAttrTypeNative:writers_passwd: ts
dsAttrTypeNative:writers_picture: ts
dsAttrTypeNative:writers_unlockOptions: ts
dsAttrTypeNative:writers_UserCertificate: ts
dsAttrTypeNative:accountPolicyData:
  <?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
  <plist version="1.0">
  <dict>
    <key>creationTime</key>
    <real>1687821212.484699</real>
    <key>failedLoginCount</key>
    <integer>0</integer>
    <key>failedLoginTimestamp</key>
    <integer>0</integer>
    <key>passwordLastSetTime</key>
    <real>1687821212.507021</real>
  </dict>
</plist>

dsAttrTypeNative:AvatarRepresentation:
dsAttrTypeNative:record_daemon_version: 8780000
dsAttrTypeNative:unlockOptions: 0
AppleMetaNodeLocation: /Local/Default
AuthenticationAuthority: ;SecureToken; ;ShadowHash;HASHLIST:<SALTED-SHA512-PBKDF2,SRP-RFC5054-4096-SHA512-PBKDF2> ;Kerberosv5;;ts@LKDC:SHA1.8DCD22811DA43DBA95A290C16E6FAF928CE94D09;
LKDC:SHA1.8DCD22811DA43DBA95A290C16E6FAF928CE94D09;
GeneratedUID: 65A093F1-FDBF-4E81-A128-942772AE4EA4
NetworkSignIn:
  2023-06-26 23:13:32 +0000
NetworkUser: ts@jamfse.io
NFSHomeDirectory: /Users/ts
OIDCProvider: Azure
Password: *****
Picture:
  /Library/User Pictures/Fun/Fortune Cookie.heic
PrimaryGroupID: 20
RealName:
  Trade Show
RecordName: ts
RecordType: dsRecTypeStandard:Users
UniqueID: 502
UserShell: /bin/zsh
ts@H2WGW2C9Q6NV ~ %
```

# dsc1

```
dsc1 . read /Users/$user
```

```
NFSHomeDirectory: /Users/ts
Password: ****
PrimaryGroupID: 20
RealName:
  Trade Show
RecordName: ts
RecordType: dsRecTypeStandard:Users
UniqueID: 502
UserShell: /bin/zsh
```



# dsc1

## Individual Keys

```
dsc1 . read /Users/$user AuthenticationAuthority
```

# dscl

## Individual Keys

```
dscl . read /Users/$user AuthenticationAuthority
```

```
~ % dscl . read /Users/$user accountPolicyData
```

```
dsAttrTypeNative:accountPolicyData:
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
  <key>creationTime</key>
  <real>1672773068.921921</real>
  <key>failedLoginCount</key>
  <integer>0</integer>
  <key>failedLoginTimestamp</key>
  <integer>0</integer>
  <key>passwordLastSetTime</key>
  <real>1682003884.02179</real>
</dict>
</plist>
```



# dscl

## Individual Keys

```
dscl . read /Users/$user AuthenticationAuthority
```

```
dscl . -readpl /Users/$user accountPolicyData creationTime
```

```
dscl . -readpl /Users/$user accountPolicyData failedLoginTimestamp
```

Dump the whole record to XML for further munging

```
dscl -plist . read /Users/$user
```

Append a record with stuff

```
dscl . -append /Users/$user Comment "User is a menace."
```

Remove keys from a record

```
dscl . delete /Users/$user Comment
```

# dsc1

```
dsc1 . read /Users/$user
```

```
dsAttrTypeNative:_writers_AvatarRepresentation: ts
dsAttrTypeNative:_writers_hint: ts
dsAttrTypeNative:_writers_jpegphoto: ts
dsAttrTypeNative:_writers_passwd: ts
dsAttrTypeNative:_writers_picture: ts
dsAttrTypeNative:_writers_unlockOptions: ts
dsAttrTypeNative:_writers_UserCertificate: ts
dsAttrTypeNative:accountPolicyData:
  <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTD
<plist version="1.0">
<dict>
  <key>creationTime</key>
  <real>1687821212.484699</real>
  <key>failedLoginCount</key>
  <integer>0</integer>
  <key>failedLoginTimestamp</key>
  <integer>0</integer>
  <key>passwordLastSetTime</key>
  <real>1687821212.507021</real>
</dict>
</plist>

dsAttrTypeNative:AvatarRepresentation:
dsAttrTypeNative:record_daemon_version: 8780000
dsAttrTypeNative:unlockOptions: 0
AppleMetaNodeLocation: /Local/Default
AuthenticationAuthority: ;SecureToken; ;ShadowHash;HASHLIST:<SALTED-SHA512-PE
LKDC:SHA1.8DCD22811DA43DBA95A290C16E6FAF928CE94D09;
GeneratedUID: 65A093F1-FDBF-4E81-A128-942772AE4EA4
NetworkSignIn:
  2023-06-26 23:13:32 +0000
NetworkUser: ts@jamfse.io
NFHomeDirectory: /Users/ts
OIDCProvider: Azure
Password: *****
Picture:
  /Library/User Pictures/Fun/Fortune Cookie.heic
PrimaryGroupID: 20
RealName:
  Trade Show
RecordName: ts
RecordType: dsRecTypeStandard:Users
UniqueID: 502
UserShell: /bin/zsh _
```



# dseditgroup

It says “edit” in the name so that must be all it does, right?

```
dseditgroup -o read admin
```

```
dsAttrTypeStandard:GroupMembership -
    root
    jamfManagement
dsAttrTypeStandard:GeneratedUID -
    ABCDEFAB-CDEF-ABCD-EFAB-CDEF00000050
dsAttrTypeStandard:RecordName -
    admin
    BUILTIN\Administrators
dsAttrTypeStandard:AppleMetaNodeLocation -
    /Local/Default
dsAttrTypeStandard:GroupMembers -
    FFFFFFFE-DDDD-CCCC-BBBB-AAAA00000000
    2C651619-AB7D-4E29-90B5-D1C817E06D24
dsAttrTypeStandard:RecordType -
    dsRecTypeStandard:Groups
dsAttrTypeStandard:SMBSID -
    S-1-5-32-544
dsAttrTypeStandard:PrimaryGroupID -
    80
dsAttrTypeStandard:RealName -
    Administrators
dsAttrTypeStandard:Password -
    *      †
```

List all local groups

```
dscacheutil -q group
```

# dseditgroup

It says “edit” in the name so that must be all it does, right?

```
dseditgroup -o read admin
```

Check if an individual user is an admin or not

```
dseditgroup -m "$user" -o checkmember admin
```

```
yes sean.rabbitt is a member of admin  
no ts is NOT a memberu of admin
```



# dseditgroup

It says “edit” in the name so that must be all it does, right?

```
dseditgroup -o read admin
```

Check if an individual user is an admin or not

```
dseditgroup -m "$user" -o checkmember admin
```

```
yes sean.rabbitt is a member of admin  
no ts is NOT a member of admin
```

```
echo "Demoting $elevateThisUser to standard account"  
/usr/sbin/dseditgroup -o edit -d "$elevateThisUser" -t user admin  
echo "Elevating $elevateThisUser to admin account"  
/usr/sbin/dseditgroup -o edit -a "$elevateThisUser" -t user admin
```

# Changing a user's local password

Or, why do I need four different ways to accomplish the same thing?

```
dscl . -passwd /Users/$user [new_password | old_password new_password]
```

```
passwd
```

```
pwpolicy -a authenticator -u user -setpassword newpassword
```

```
sysadminctl -newPassword <new password> -oldPassword <old password> [-passwordHint <password hint>]
```

```
sysadminctl -resetPasswordFor <local user name>  
-newPassword <new password>  
[-passwordHint <password hint>]  
(interactive) || -adminUser <administrator user name> -adminPassword <administrator password>)
```



# sysadminctl

The command line tool that gets jammed full of stuff when nobody knows where else to put it.

- User - Create / Delete
- Password - Set / Force Reset
- FileVault secure token - Enable / Disable / Status
- Auto-login - Enable / Disable / Status
- Guest accounts - Enable / Disable / Status
- Samba (SMB) or Apple Filing Protocol (AFP) guest access - Enable / Disable / Status

# sysadminctl

The command line tool that gets jammed full of stuff when nobody knows where else to put it.

- User - Create / Delete
- Password - Set / Force Reset
- FileVault secure token - Enable / Disable / Status
- Auto-login - Enable / Disable / Status
- Guest accounts - Enable / Disable / Status
- Samba (SMB) or Apple Filing Protocol (AFP) guest access - Enable / Disable / Status
- Automatic Time (?!?) - Enable / Disable / Status (but not which NTP server, thats in /etc/ntp.conf)
- File System encryption - Status
- Screen Lock - Status OR disable / seconds to enable with local admin password required



# pwdpolicy

Wait, it does more than reset passwords?

```
pwdpolicy -a authenticator -u user -setpassword newpassword
```

Disable a local user from logging in

```
pwdpolicy -u user -disableuser
```

```
pwdpolicy -u user -enableuser
```

Do something terrible and set a local account policy manually

```
pwdpolicy -u user -setpolicy "minChars=4 maxFailedLoginAttempts=3"
```

Clear account policies (aka set it back to 4 character minimum requirement)

```
pwdpolicy -clearaccountpolicies
```

# pwdpolicy

Wait, it does more than reset passwords?

```
pwdpolicy -a authenticator -u user -setpassword newpassword
```

Disable a local user from logging in

```
pwdpolicy -u user -disableuser
```

```
pwdpolicy -u user -enableuser
```

Do something terrible and set a local account policy manually

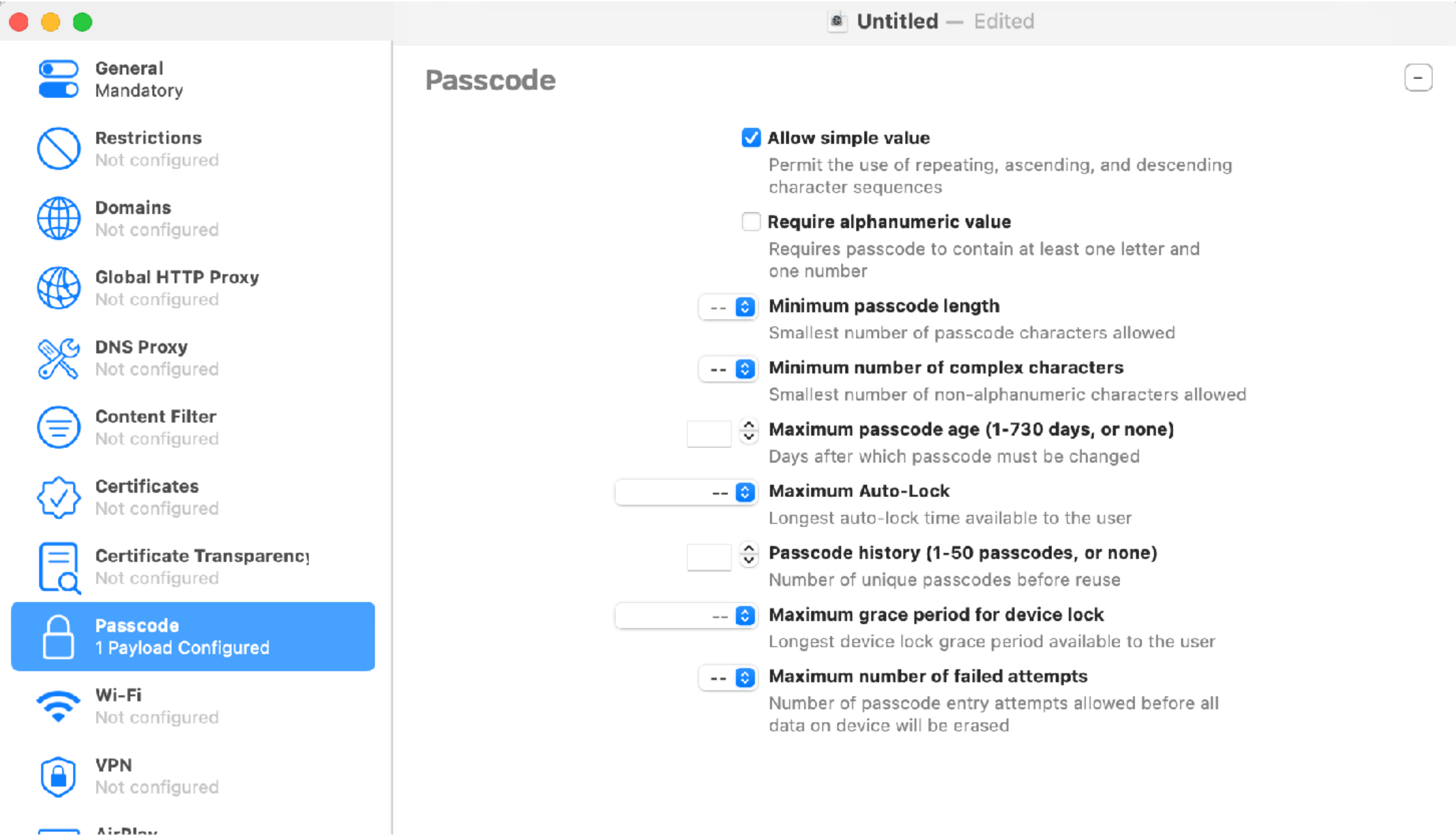
```
pwdpolicy -u user -setpolicy "minChars=4 maxFailedLoginAttempts=3"
```

Clear account policies (aka set it back to 4 character minimum requirement)

```
pwdpolicy -clearaccountpolicies *
```



# Pushing settings via MDM...



# Pushing settings via MDM...

Unscoping or removing a profile  
does not remove the password policy  
from the device.

```
pwpolicy -clearaccountpolicies
```

Computers : Configuration Profiles

← New macOS Configuration Profile

Options

Scope

Search...

Login Items  
Not configured

Login Window  
Not configured

Managed Login Items  
Not configured

Mobility  
Not configured

Network  
Not configured

Notifications  
Not configured

Parental Controls  
Not configured

Passcode  
Not configured

Printing  
Not configured

Privacy Preferences Policy Control  
Not configured

Proxies  
Not configured

Restrictions  
Not configured

SCEP  
Not configured

Security and Privacy  
Not configured

Single Sign-On Extensions  
Not configured

Smart Card  
Not configured

Software Update  
Not configured

System Extensions  
Not configured

System Migration  
Not configured

Time Machine  
Not configured

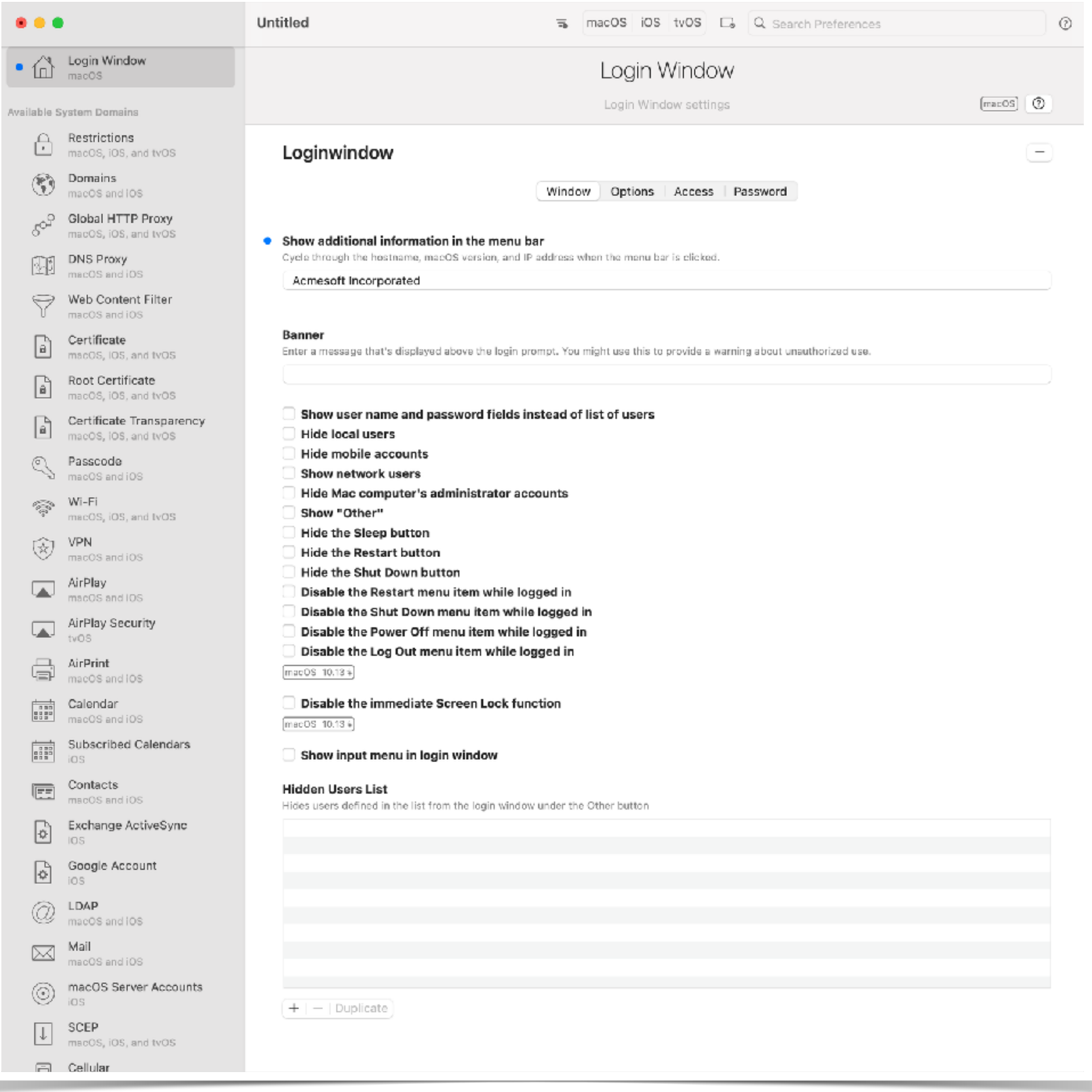
Passcode

Specify passcode policies. Only the included settings will be enforced on the computers in scope.

Exclude all

Setting	Include
Require Passcode Enforce setting passcode on the computer.	<input checked="" type="checkbox"/>
Complex Passcode Passcode cannot contain repeating, ascending, and descending character sequences.	<div>Enforce Ignore</div> <input checked="" type="checkbox"/>
Alphanumeric Value Passcode must contain at least one letter and one number.	<div>Enforce Ignore</div> <input checked="" type="checkbox"/>
Minimum Passcode Length Smallest number of passcode characters allowed	<input checked="" type="checkbox"/>
Minimum Number of Complex Characters Smallest number of non-alphanumeric characters allowed	<input checked="" type="checkbox"/>
Maximum Passcode Age Number of days until the passcode must be changed (1-730)	<input checked="" type="checkbox"/>
Passcode History Number of unique passcodes before reuse (1-50)	<input checked="" type="checkbox"/>
Maximum Auto-Lock Number of minutes before the computer automatically locks	<input checked="" type="checkbox"/>
Maximum Grace Period for Computer Lock Period of inactivity before the passcode is required to unlock the computer	<input checked="" type="checkbox"/>
Maximum Number of Failed Attempts Number of passcode entry attempts allowed before the computer is locked	<input checked="" type="checkbox"/>
Delay after Failed Login Attempts (Not compatible with macOS 10.11.0) Delay after maximum number of failed attempts, in minutes. Requires configuring Maximum Number of Failed Attempts.	<input checked="" type="checkbox"/>
Change at Next Authentication (macOS 10.13 or later) Force password reset on next user authentication.	<div>Enforce Ignore</div> <input checked="" type="checkbox"/>





- Login Window  
macOS
- Available System Domains
- Restrictions

macOS, iOS, and tvOS
- Domains

macOS and iOS
- Global HTTP Proxy

macOS, iOS, and tvOS
- DNS Proxy

macOS and iOS
- Web Content Filter

macOS and iOS
- Certificate

macOS, iOS, and tvOS
- Root Certificate

macOS, iOS, and tvOS
- Certificate Transparency

macOS, iOS, and tvOS
- Passcode

macOS and iOS
- Wi-Fi

macOS, iOS, and tvOS
- VPN

macOS and iOS
- AirPlay

macOS and iOS
- AirPlay Security

tvOS
- AirPrint

macOS and iOS
- Calendar

macOS and iOS
- Subscribed Calendars

iOS
- Contacts

macOS and iOS
- Exchange ActiveSync

iOS
- Google Account

iOS
- LDAP

macOS and iOS
- Mail

macOS and iOS
- macOS Server Accounts

iOS
- SCEP

macOS, iOS, and tvOS
- Cellular

- Login Window
- Show additional information in the menu bar
- Acme
- Banner
- Enter a message to display above the login prompt.
- Show additional information in the menu bar
- Hide additional information in the menu bar
- Hide additional information in the menu bar
- Hide additional information in the menu bar
- Hide additional information in the menu bar
- Hide additional information in the menu bar
- Hide additional information in the menu bar
- Hide additional information in the menu bar
- Hide additional information in the menu bar
- Hide additional information in the menu bar
- Hide additional information in the menu bar
- Hidden Users
- Hides users from the login window.
- +
- 
- 0

Computers : Configuration Profiles

← A Test Login Window Policy

Options Scope

Search...

Login Window

Window Options Access Script

☒ Show additional information in the menu bar

Show the host name, macOS version and IP address when the menu bar is clicked.

Banner

A message displayed above the login prompt.

macOS has a built in screen reader called VoiceOver

Login Prompt

The display style and related options of the login prompt.

☐ Name and password text fields

☒ List of users able to use these computers

☒ Show local users

☒ Show mobile accounts

☐ Show network users

☒ Show computer's administrators

☒ Show "Other..."

☐ Show Shut Down button

Managed Login Items

Not configured

Mobility

Not configured

Network

Not configured

Notifications

Not configured

Parental Controls

Not configured

Passcode

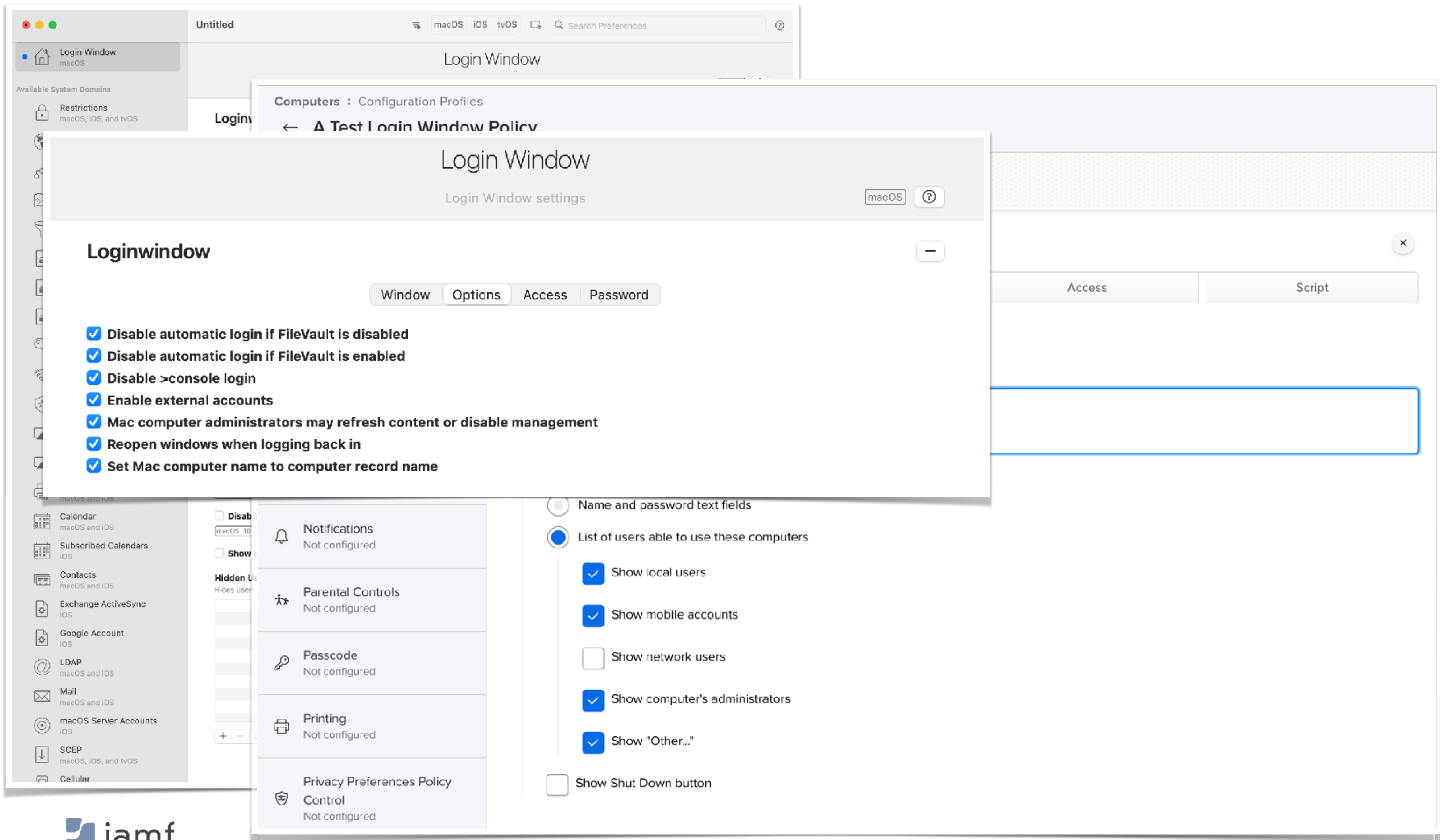
Not configured

Printing

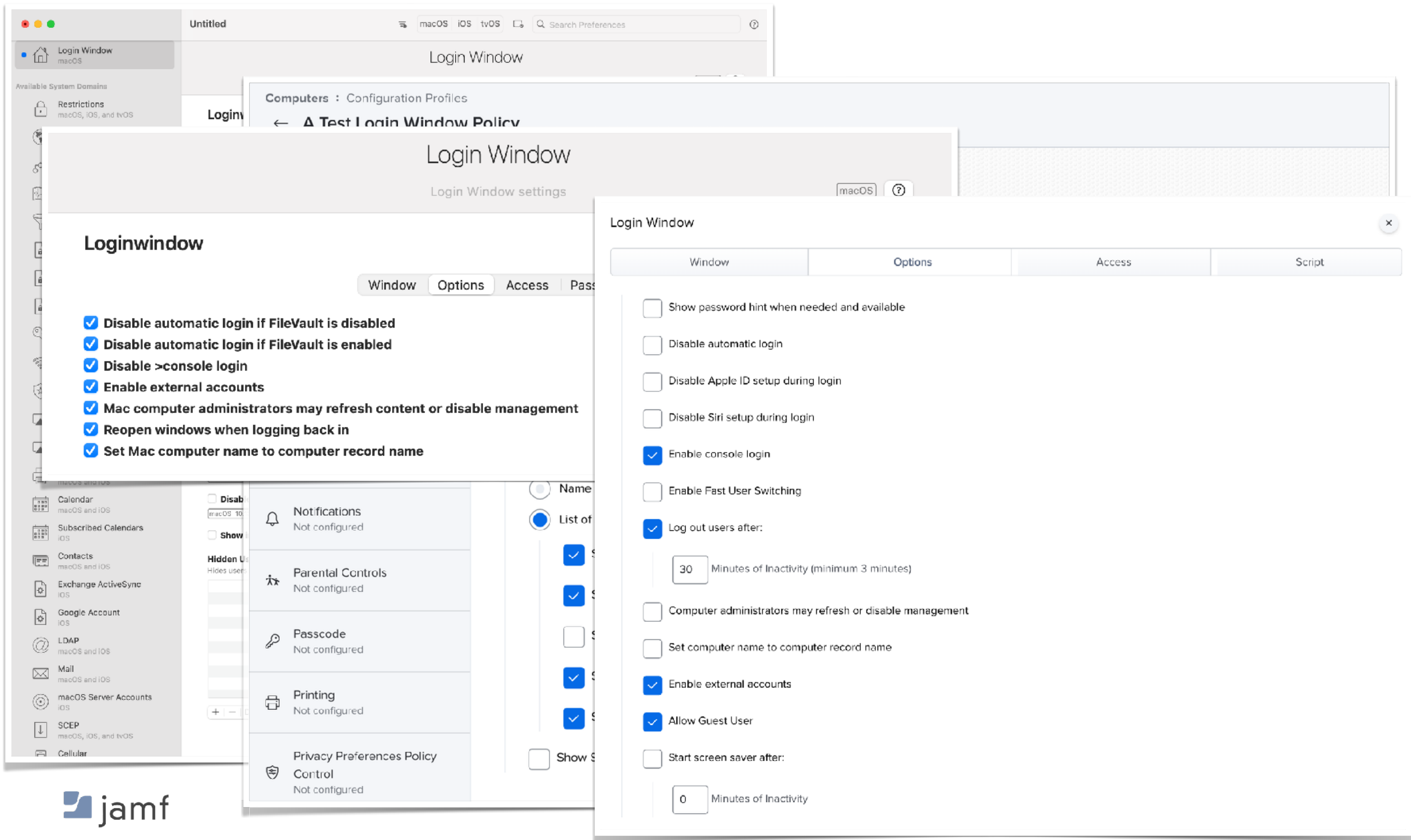
Not configured

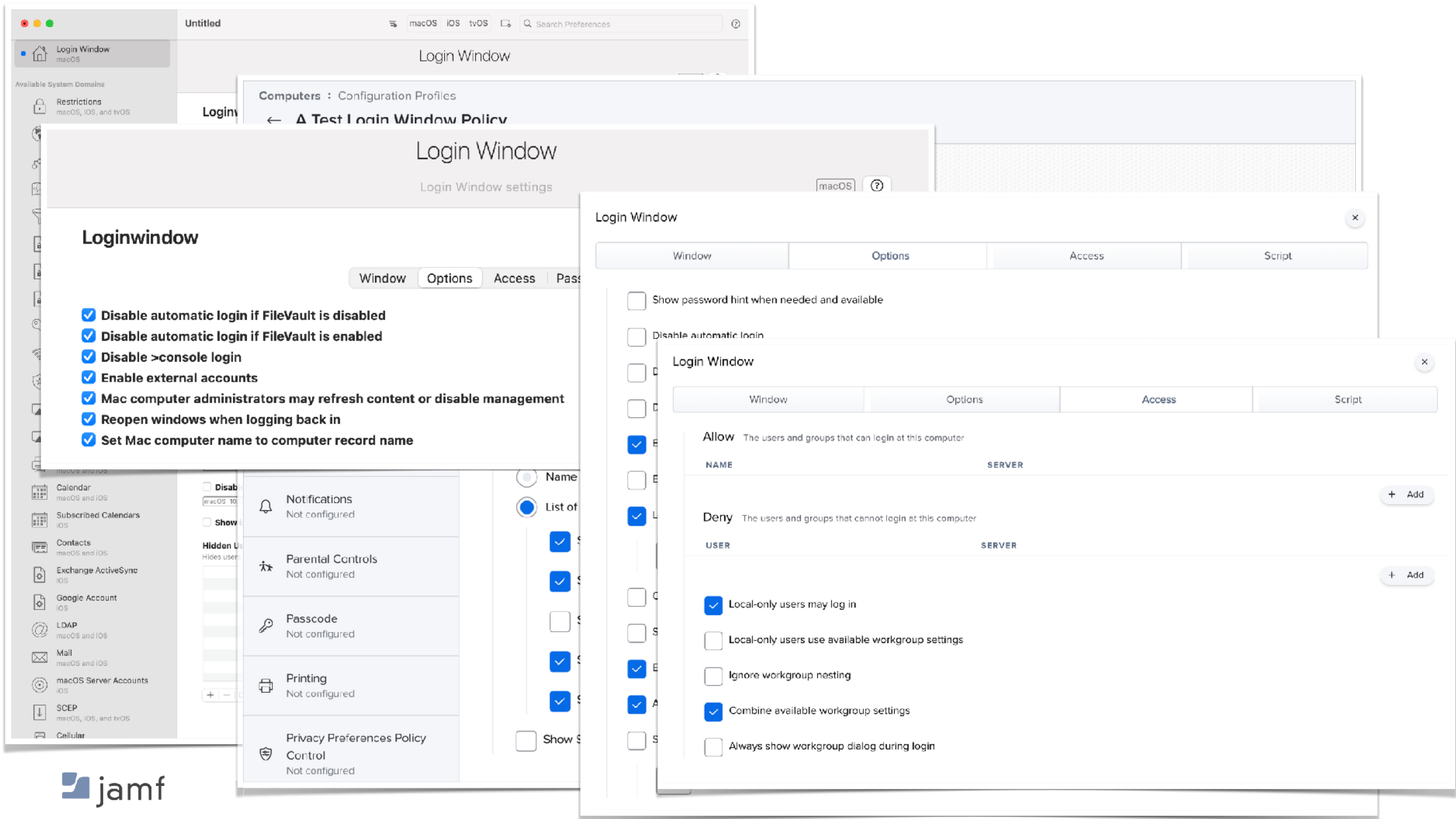
Privacy Preferences Policy Control

Not configured









# Restrictions

Use this section to configure restrictions on a device.

macOS

iOS

tvOS



## Restrictions



General

AirDrop

AirPlay

AirPrint

Apps

Classroom

iCloud

Media

Passwords / Unlock

Siri

Updates

### ☒ Allow modifying passcode

Supervised only iOS 9.0 ↕

#### ☒ Allow modifying Touch ID / Face ID

Supervised only iOS 9.0 ↕

### ☒ Allow Touch ID / Face ID to unlock device

macOS 10.12.4 ↕ iOS 7.0 ↕

### ☒ Allow password autofill

Supervised only macOS 10.14 ↕ iOS 12.0 ↕

### ☒ Allow Apple Watch to auto unlock device

macOS 10.12 ↕ iOS 14.5 ↕

### ☒ Allow proximity based password sharing requests

Supervised only macOS 10.14 ↕ iOS 12.0 ↕ tvOS 12.0 ↕

### ☒ Allow password sharing

Supervised only macOS 10.14 ↕ iOS 12.0 ↕

### Enforced Fingerprint Timeout

Period of time in seconds after which the device will require entry of password or passcode to unlock.

macOS 12.0 ↕ iOS 15.0 ↕

### ☒ Allow Automatic Screen Saver

tvOS 15.4 ↕



Options

Scope

Search...



Printing  
Not configured



Privacy Preferences Policy  
Control  
Not configured



Proxies  
Not configured



Restrictions  
Not configured



SCEP  
Not configured



Security and Privacy  
Not configured

General

FileVault

Firewall



Single Sign-On Extensions  
Not configured



Smart Card  
Not configured



Software Update  
Not configured



System Extensions  
Not configured

# Security and Privacy: General

Only the included settings will be enforced on the devices in scope.

Exclude all

Filter:

Configured

Include

## Password Change

Restrict this setting to prevent the user from changing the password. macOS 10.10 or later

Restrict

Allow



## Set Lock Message

Restrict this setting to prevent the user from changing the Lock message. macOS 10.10 or later

Restrict

Allow



## Send diagnostic and usage data to Apple, and sharing crash data and statistics with app developers

Restrict this setting to prevent the computer from automatically submitting diagnostic reports to Apple. macOS 10.13 or later

Restrict

Allow



## Unlock macOS computer using an Apple Watch with watchOS 3 or later

Restrict this setting to disallow auto unlock. macOS 10.12 or later

Restrict

Allow



## Require Passcode to Unlock Screen

Time to delay before the password will be required to unlock or stop the screen saver

Immediately



## Gatekeeper

Allow apps downloaded from:



Mac App Store



Mac App Store and identified developers



Anywhere



Temporarily overriding the Gatekeeper setting by control-clicking to install any app

Restrict

Allow

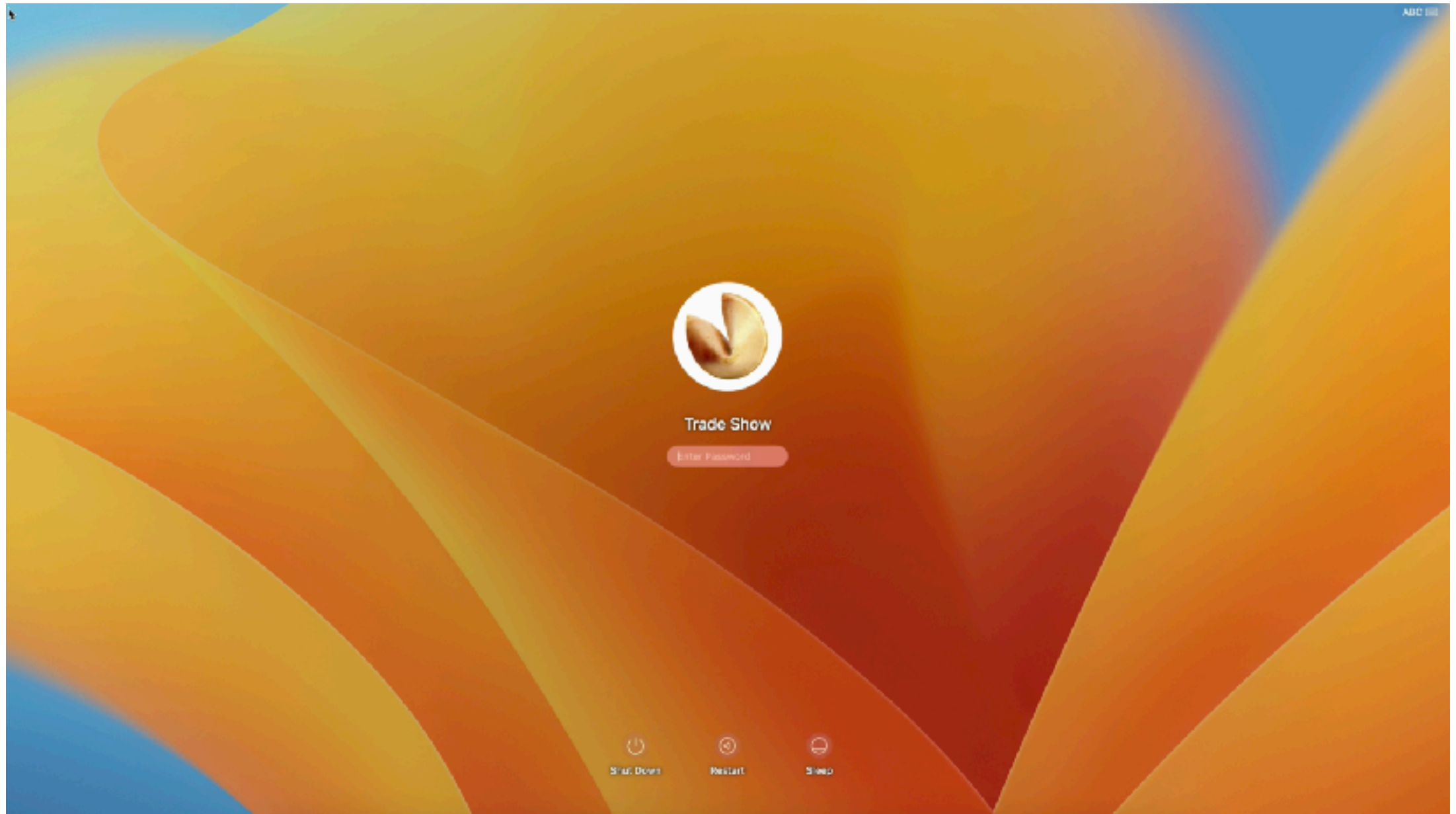
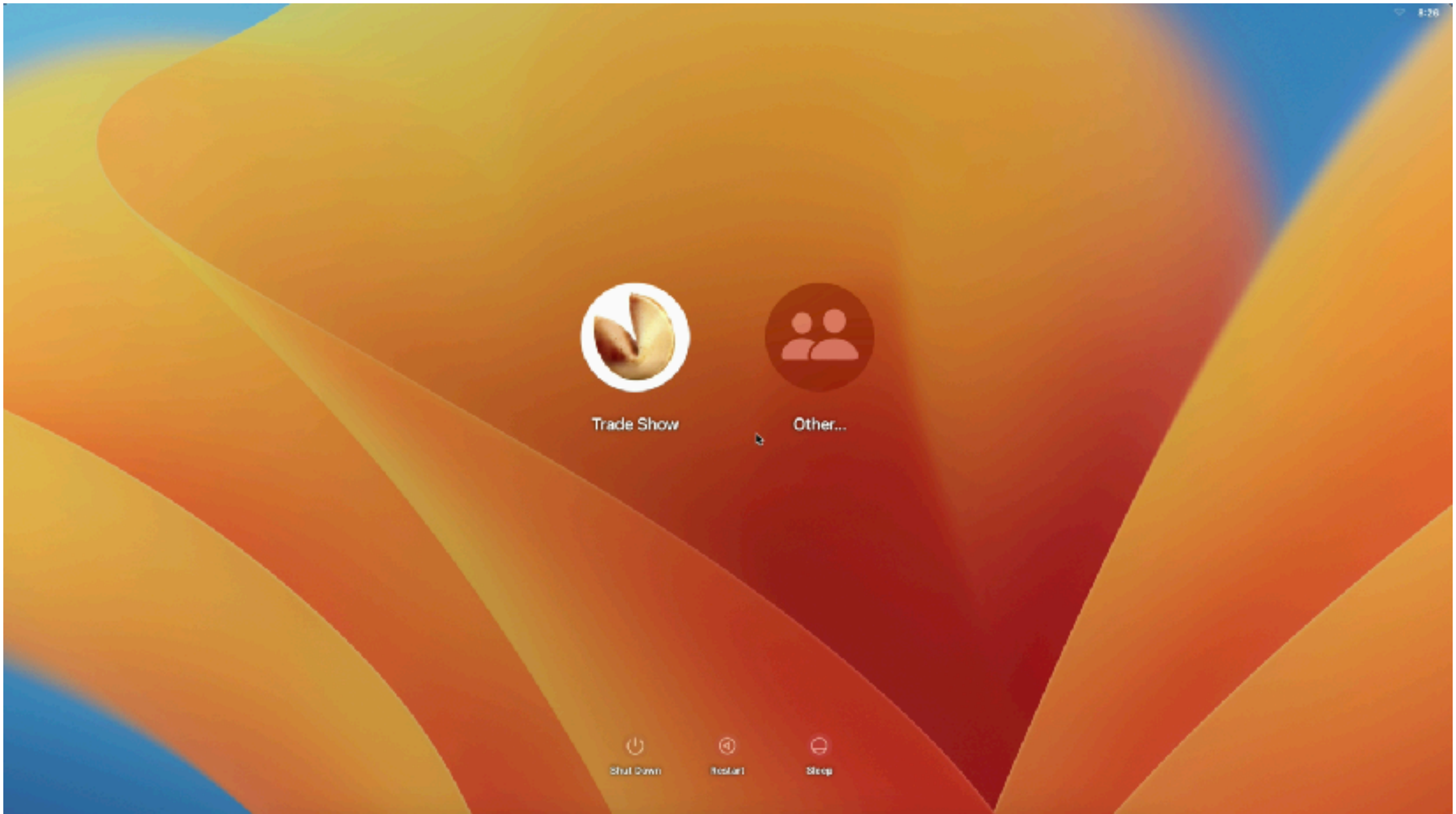
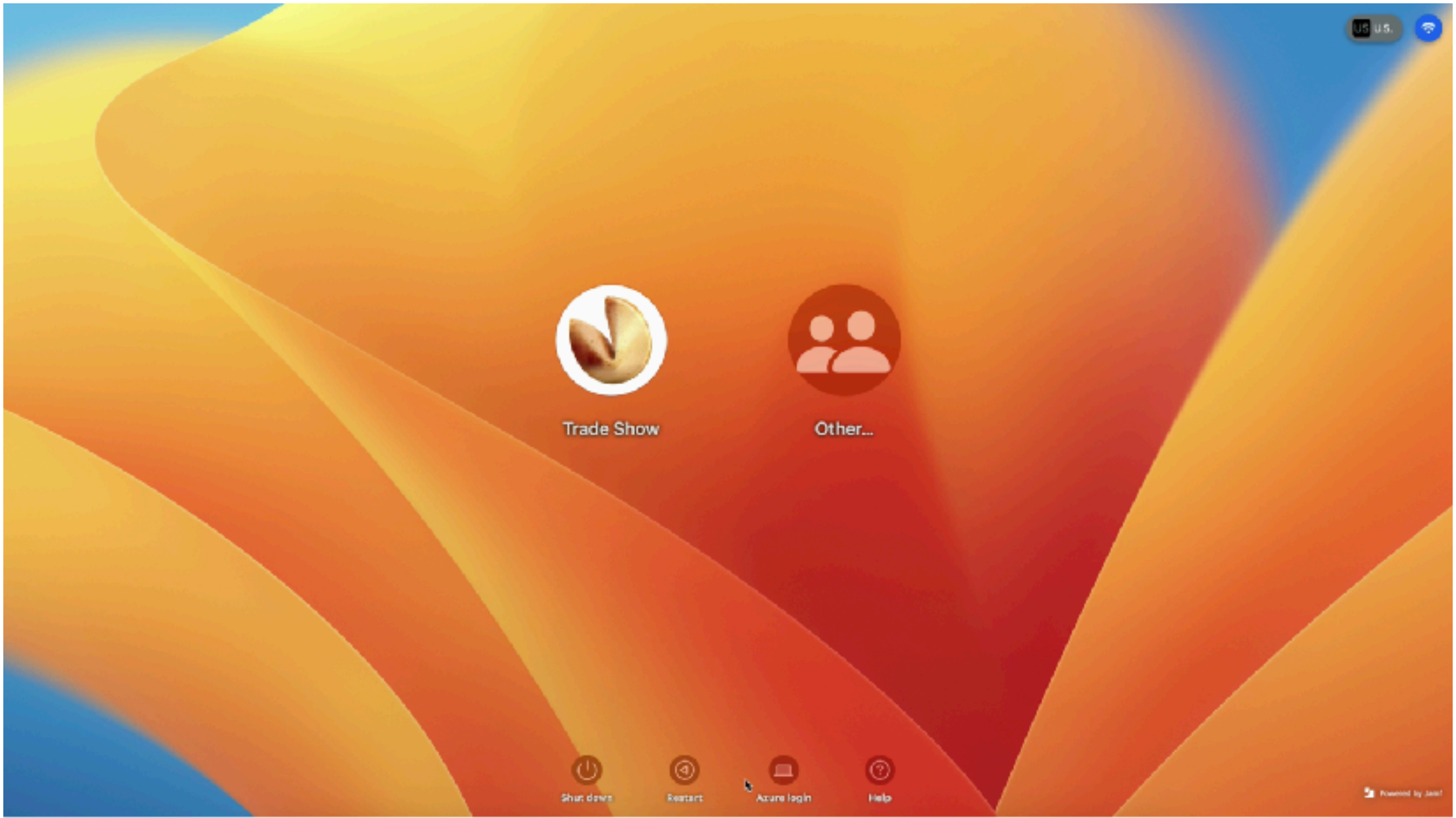
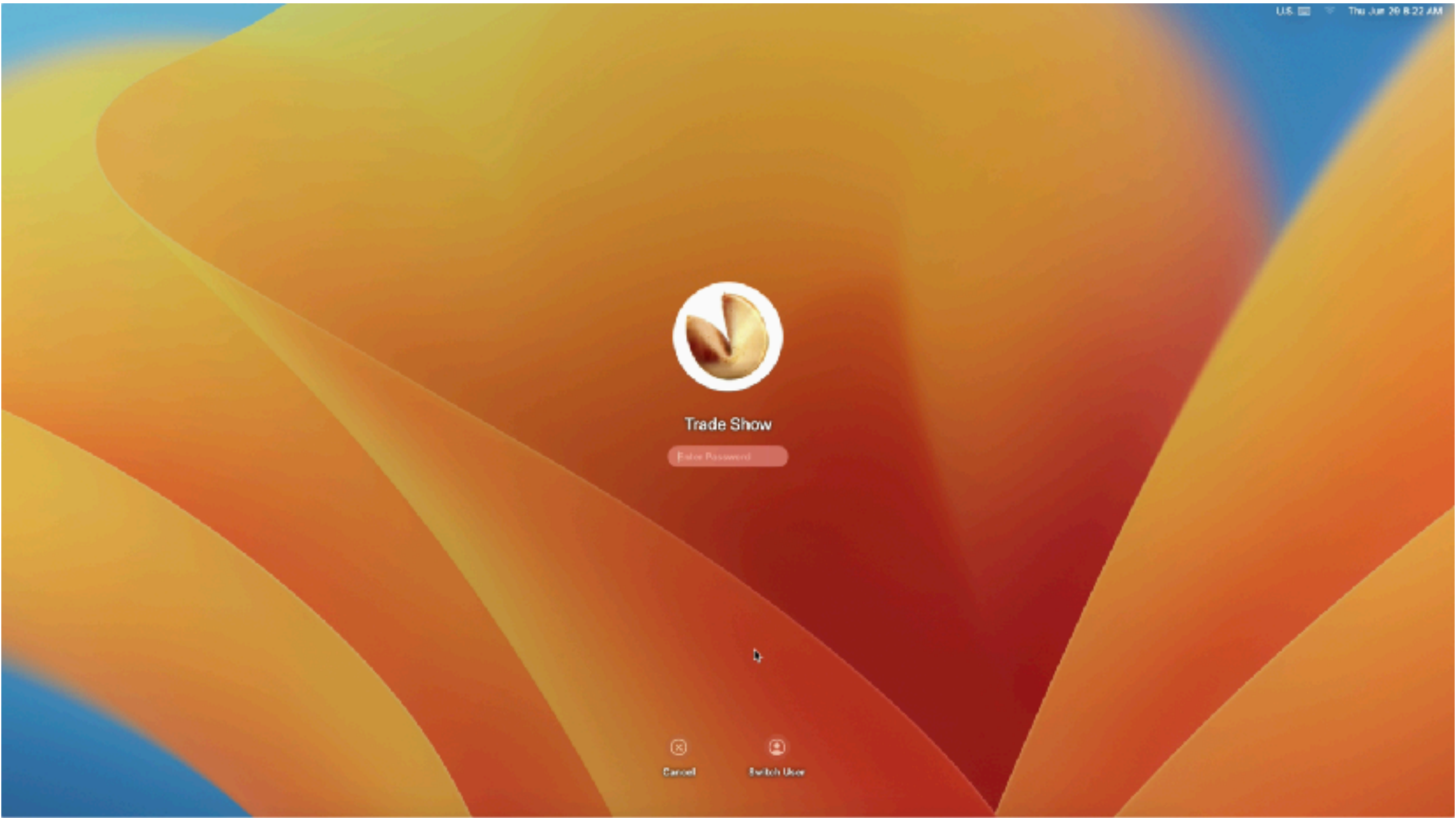


# Local User Accounts - Section Summary

- macOS is UNIX
- Useful commands
  - `dscl`
  - `dseditgroup`
  - `passwd`
  - `pwpolicy`
  - `sysadminctl`
- Unscoping a config profile donna  
undo a `pwpolicy` applied to machine
- There are a billion config profile keys  
spread across a billion payloads

**And now  
for something  
completely different.**









Trade Show

Enter Password



Shut Down



Restart

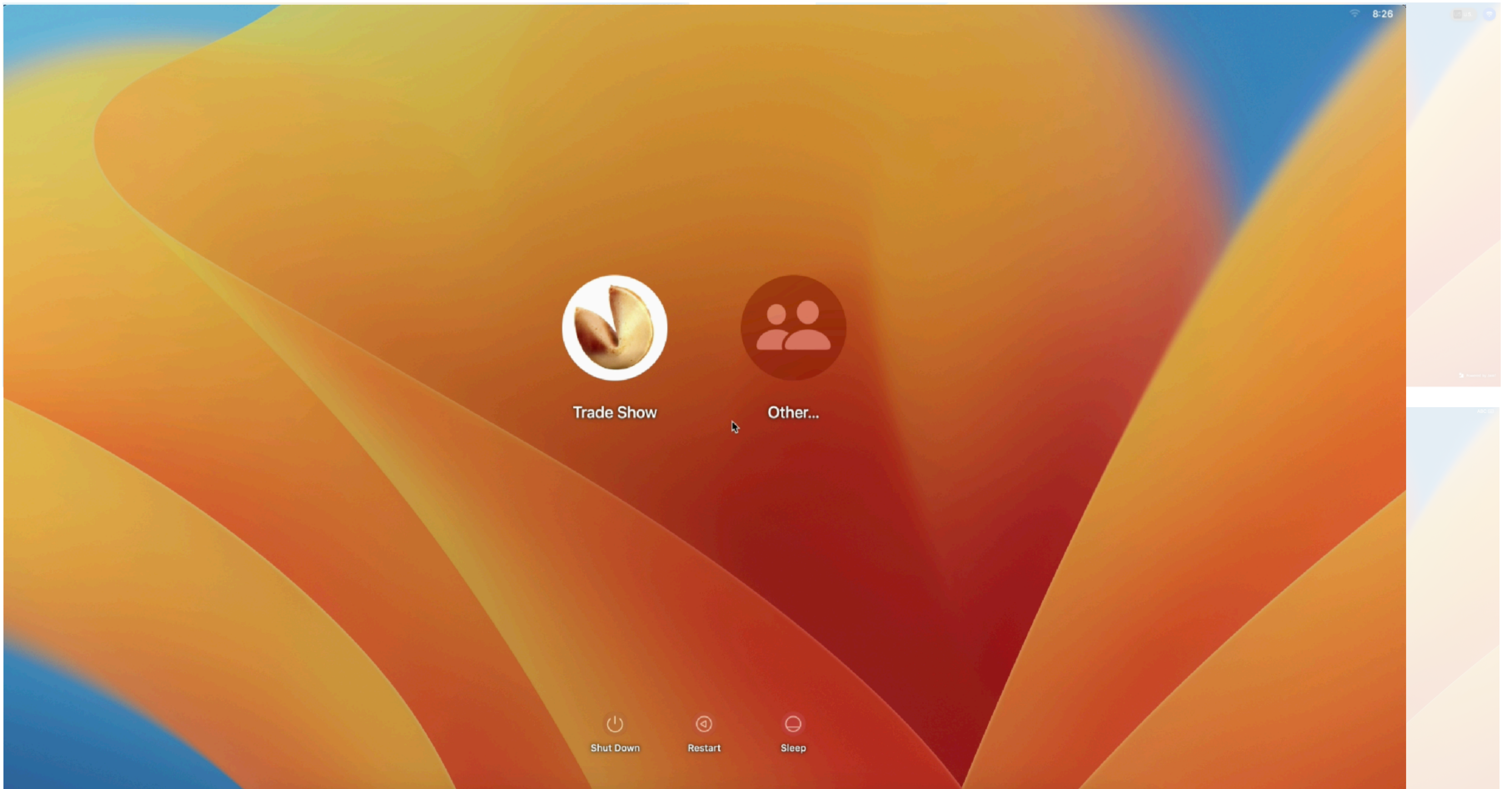


Sleep

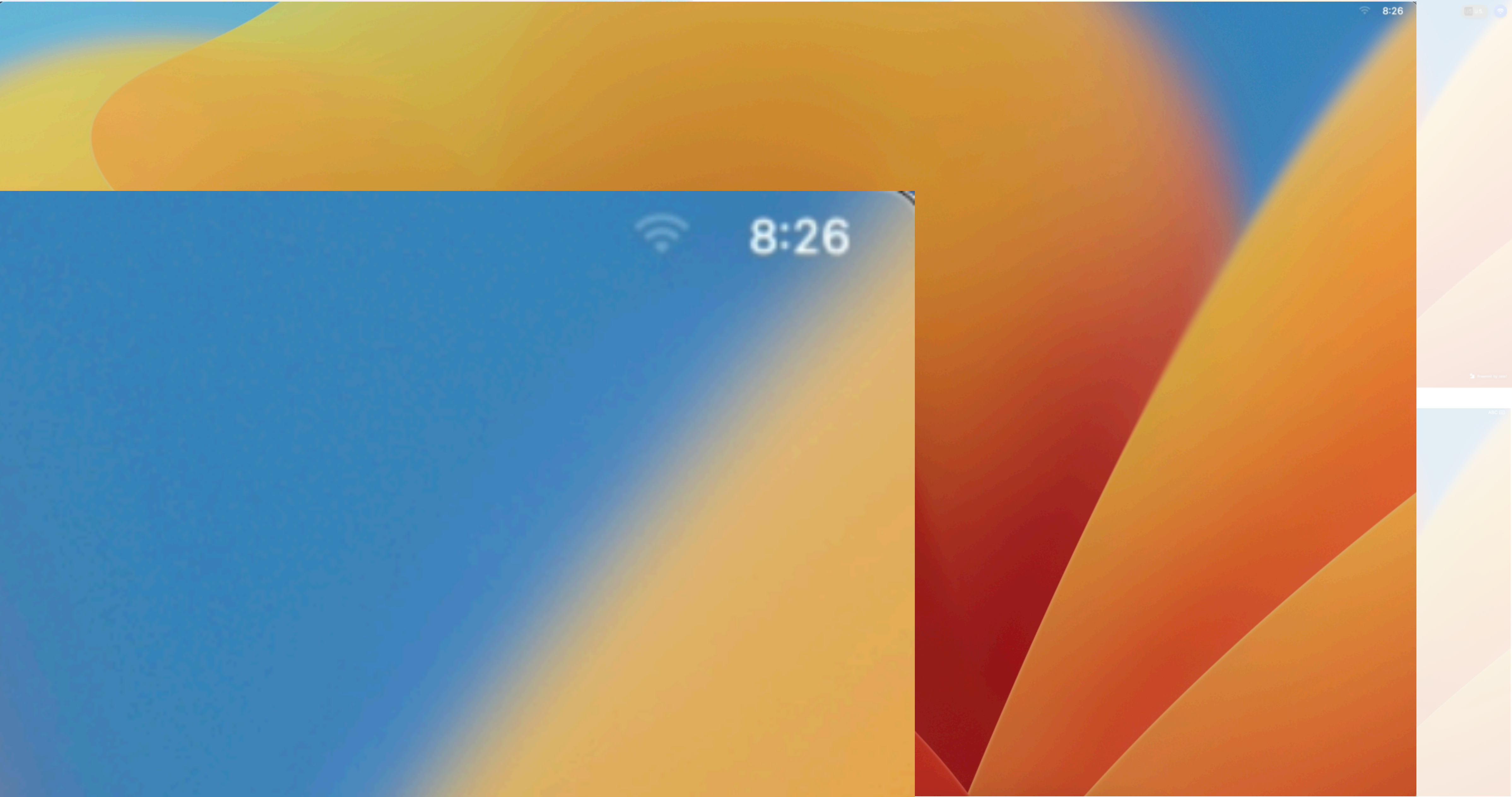
















Trade Show

Enter Password

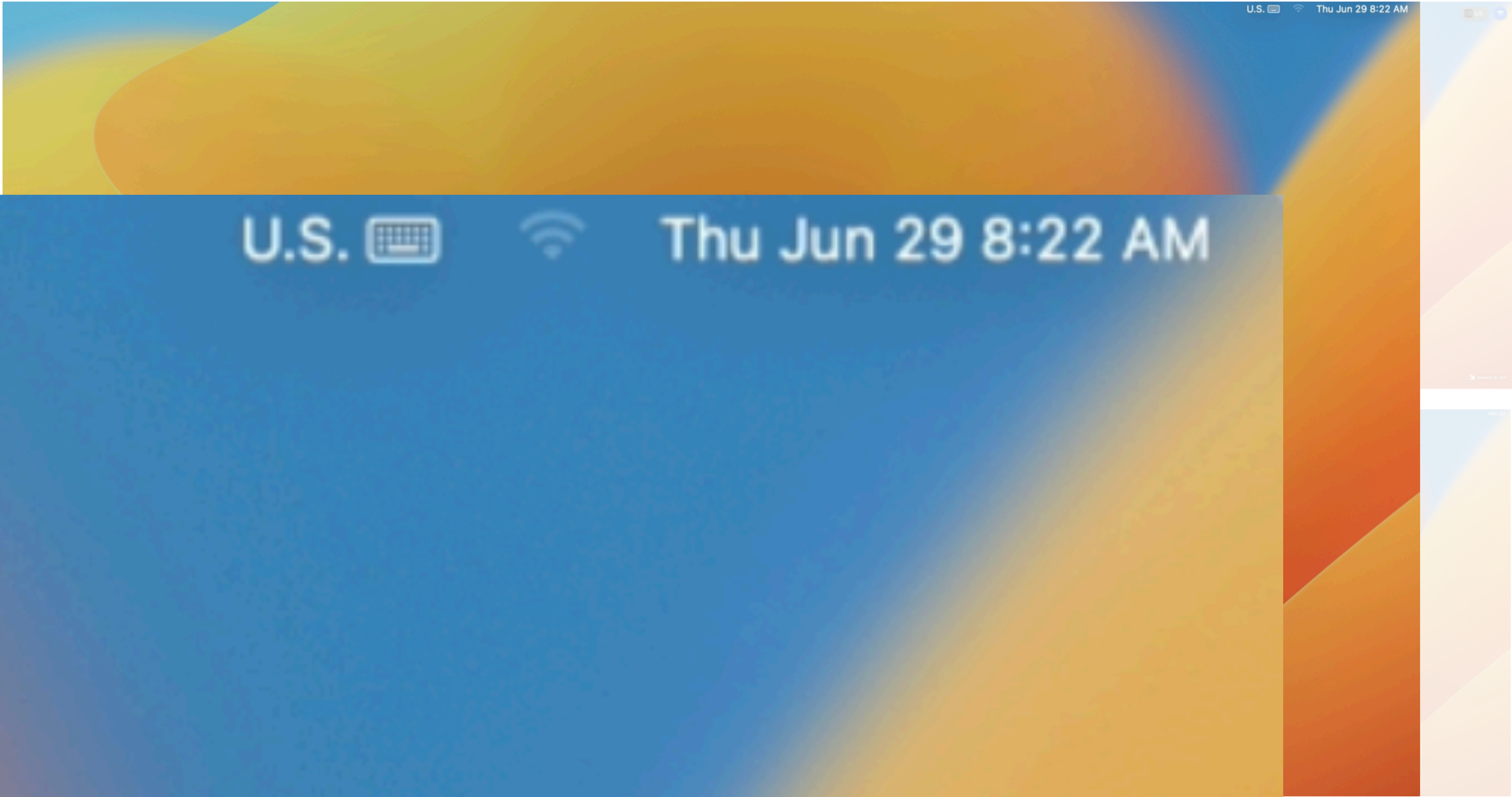


Cancel

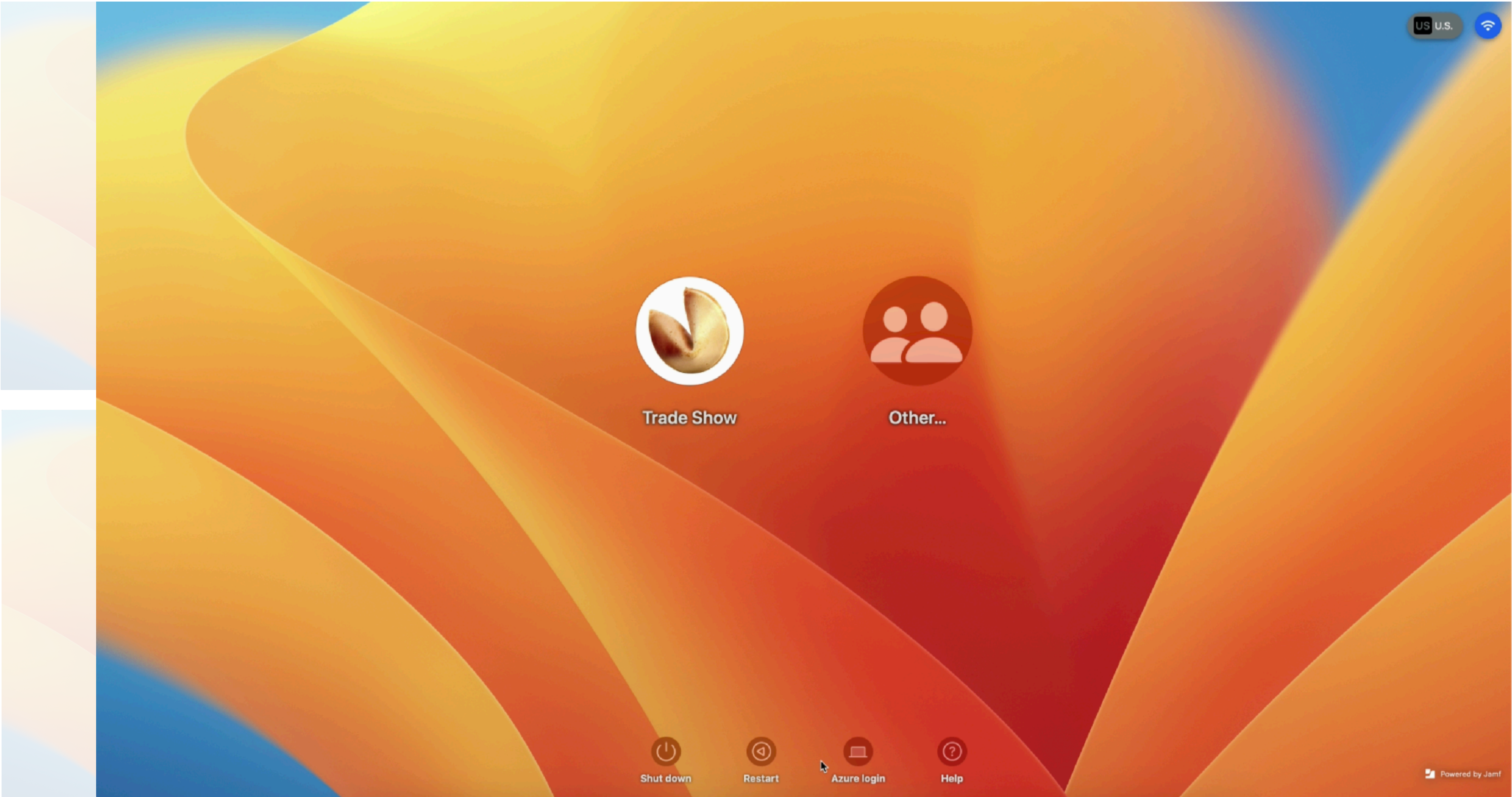


Switch User



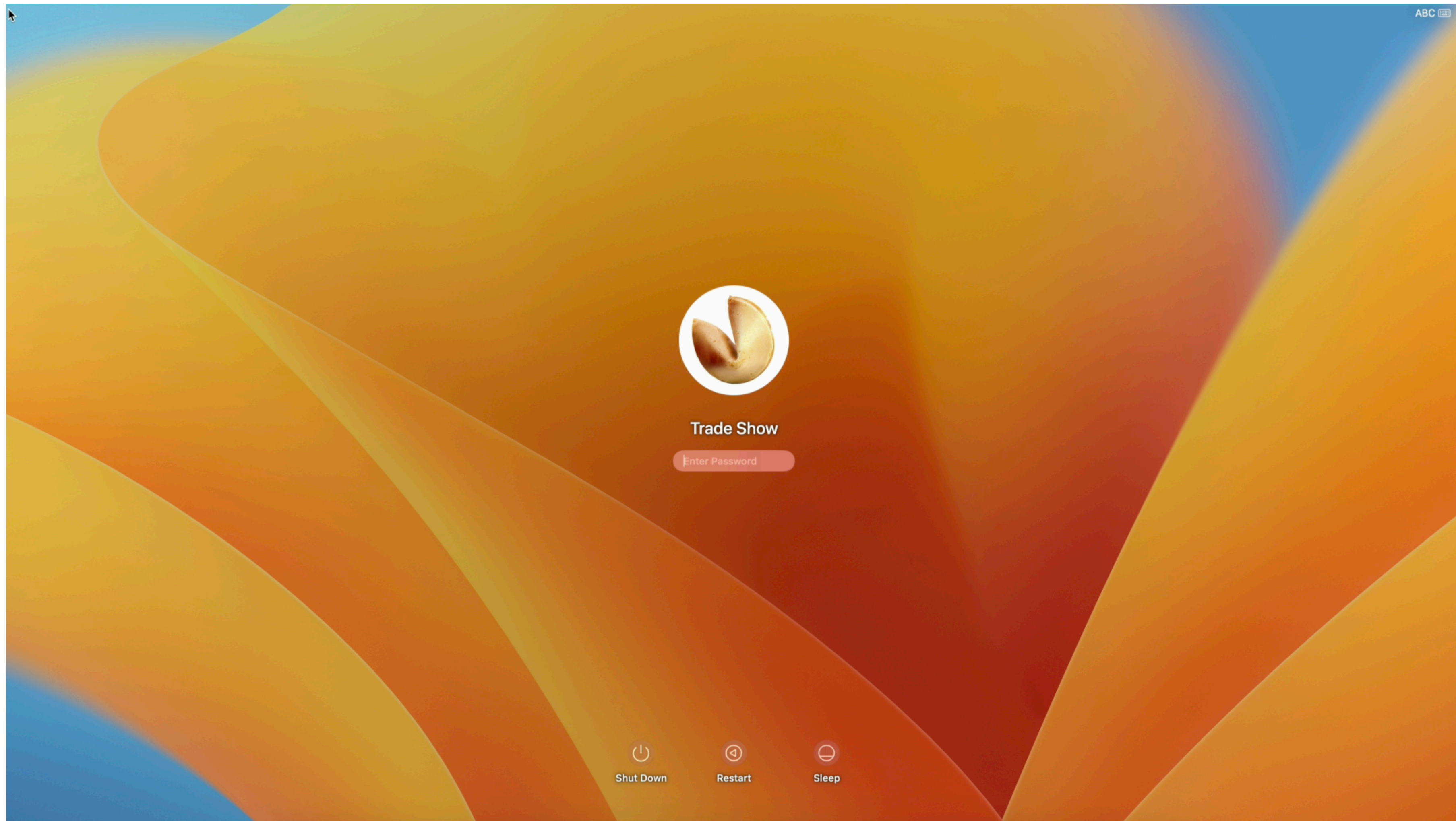






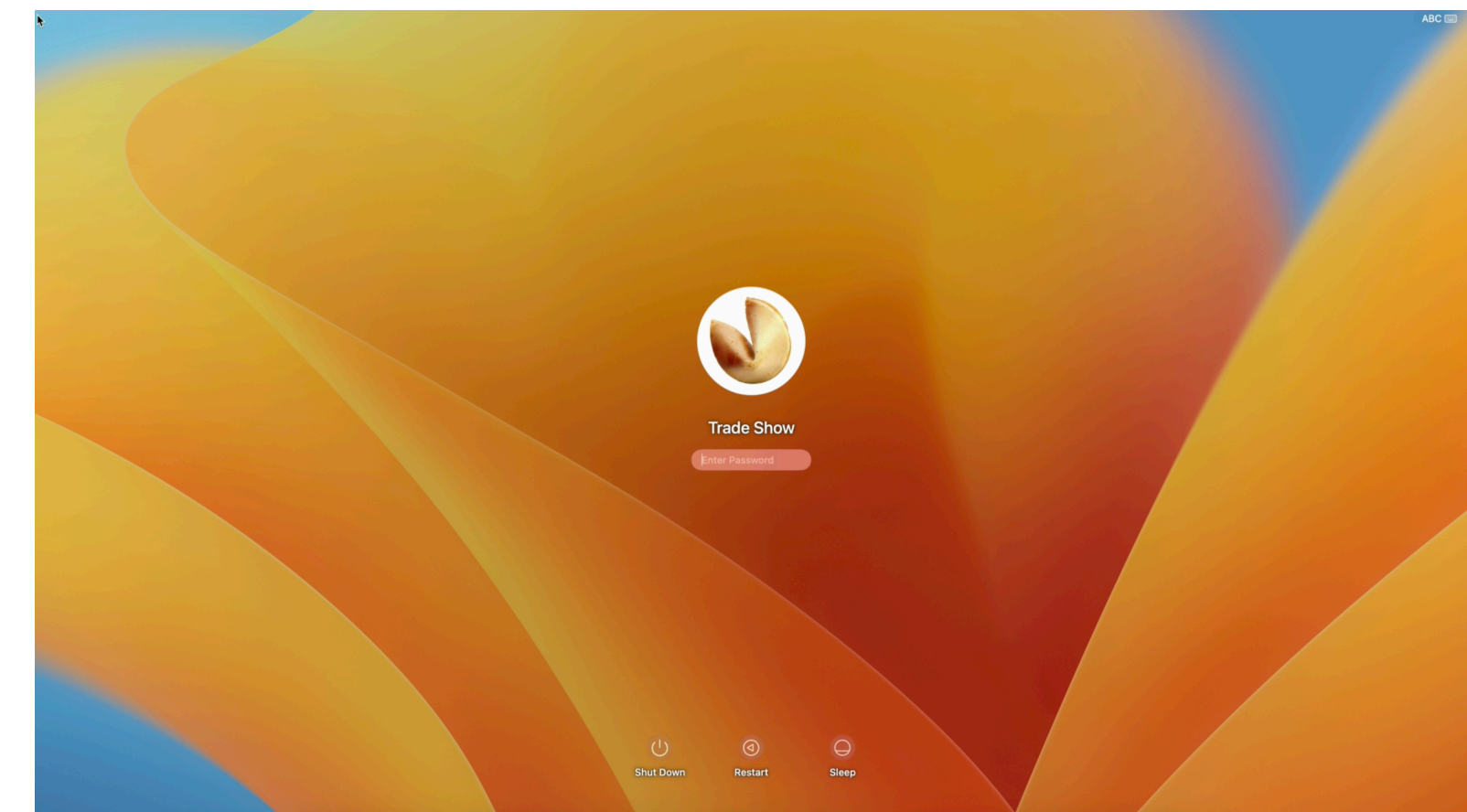
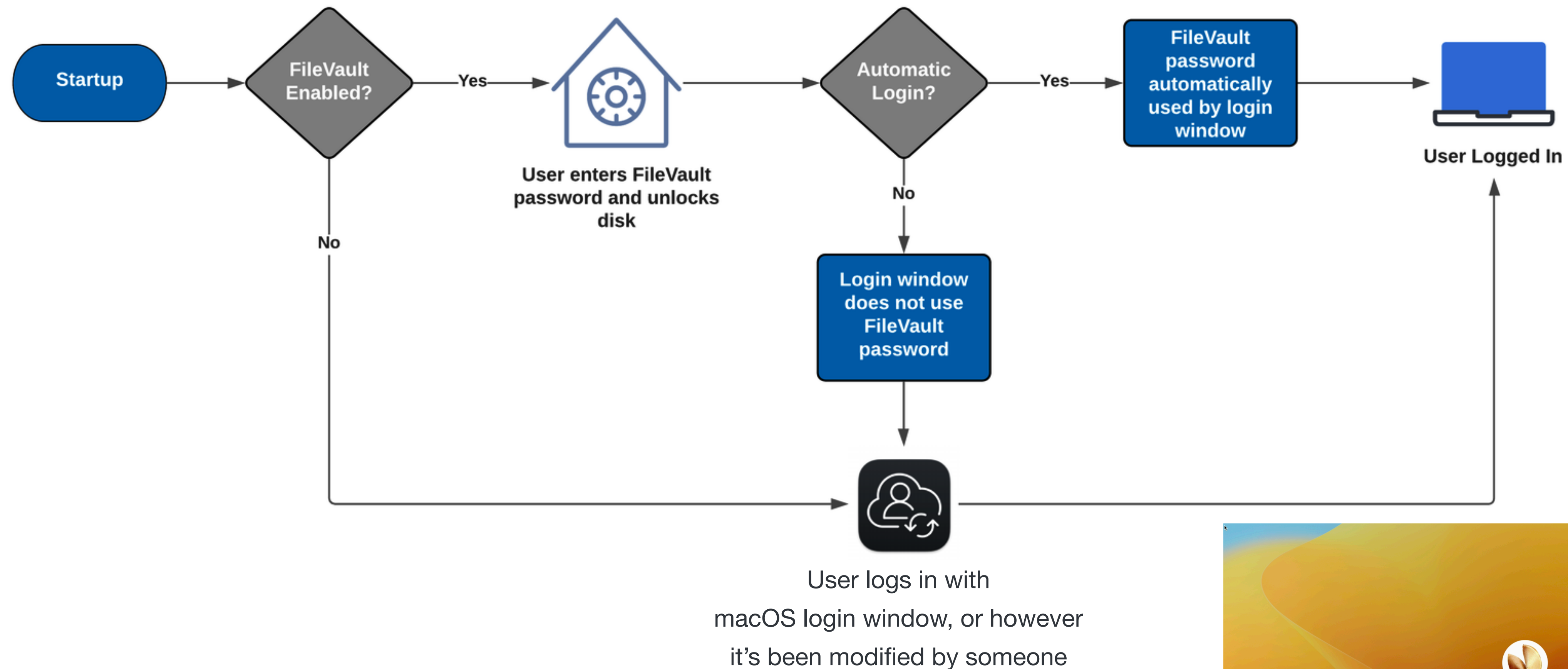


# FileVault, or why you will have a local user account forever





# FileVault, or why you will have a local user account forever



# HCS Technology Group - Resync FileVault Passwords



<https://hcsonline.com/support/blog/entry/how-to-fix-out-of-sync-filevault-password>



# Apple - Resetting a local user password



<https://support.apple.com/en-us/HT202860>

# Apple - Resetting a local user password



## Option 3: Reset using your recovery key

1. Click the option to reset using your recovery key.
2. Enter your FileVault recovery key. It's the long string of letters and numbers you received when you turned on FileVault and chose to create a recovery key instead of allowing your iCloud account (Apple ID) to unlock your disk.
3. Enter your new password information, then click Reset Password.

<https://support.apple.com/en-us/HT202860>



# On-Premises and Cloud Directory Services

<!--content warning-->

<rant>



# On-Premises Directory - Binding your Mac to AD

- Centralized account management
  - Unified password complexity policies
  - Common credentials for all on-premises services

# On-Premises Directory - Binding your Mac to AD

- Centralized account management
  - Unified password complexity policies
  - Common credentials for all on-premises services
- User and Machine based certificates
  - Key Distribution Server (KDS) on prem
  - Kerberos ticket for accessing resources



# On-Premises Directory - Binding your Mac to AD

- Centralized account management
  - Unified password complexity policies
  - Common credentials for all on-premises services
- User and Machine based certificates
  - Key Distribution Server (KDS) on prem
  - Kerberos ticket for accessing resources
- Mount and traverse DFS namespace
  - Automatic mounting of underlying SMB shares

# On-Premises Directory - Binding your Mac to AD

- All users are “Network” accounts



# On-Premises Directory - Binding your Mac to AD

- All users are “Network” accounts
- Some users can be “Mobile” accounts

# On-Premises Directory - Binding your Mac to AD

- All users are “Network” accounts
- Some users can be “Mobile” accounts
- But everyone is still also a local account, so....



# On-Premises Directory - Binding your Mac to AD

- All users are “Network” accounts
- Some users can be “Mobile” accounts
- But everyone is still also a local account, so....

**This will be a problem for you,  
guaranteed,  
every time.**

# On-Premises Directory - Binding your Mac to AD

```
dscl . list /Users OriginalNodeName
```



# On-Premises Directory - Binding your Mac to AD

```
dscl . list /Users OriginalNodeName
```

```
dscl . read /Users/$USER AuthenticationAuthority
```

# On-Premises Directory - Binding your Mac to AD

```
dscl . list /Users OriginalNodeName
```

```
dscl . read /Users/$USER AuthenticationAuthority
```

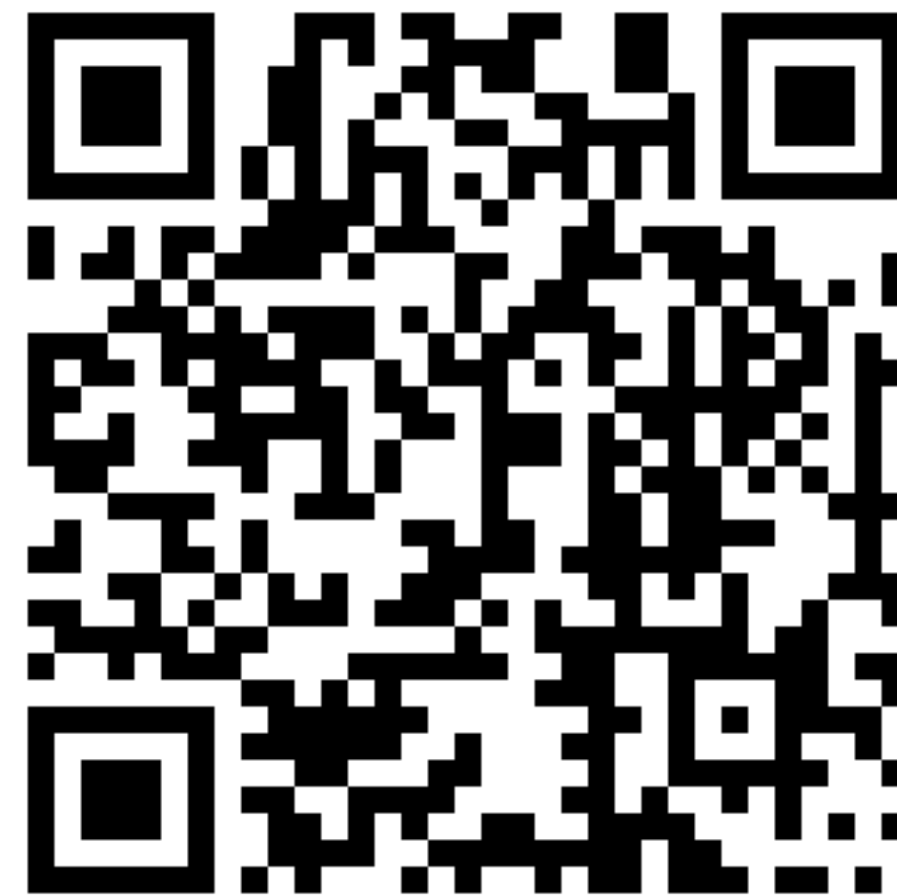
A Mobile account is just a local user account where the password happens to be the same... until it's not.



# On-Premises Directory - Binding your Mac to AD



<https://developer.apple.com/videos/play/wwdc2021/10130/>



<https://developer.apple.com/videos/play/wwdc2022/10045/>



<https://developer.apple.com/videos/play/wwdc2020/10639/>

# On-Premises Directory - Binding your Mac to AD

**DO NOT BIND MACS TO A DIRECTORY.**



<https://developer.apple.com/videos/play/wwdc2021/10130/>



<https://developer.apple.com/videos/play/wwdc2022/10045/>



<https://developer.apple.com/videos/play/wwdc2020/10639/>



</rant>

# On-Premises Directory - Alternatives

- Kerberos Single Sign-On Extension \*
  - Built into the operating system, no companion app needed
  - Configured and deployed with MDM config profiles
  - Supported by AppleCare

# On-Premises Directory - Alternatives

- Kerberos Single Sign-On Extension \*
  - Built into the operating system, no companion app needed
  - Configured and deployed with MDM config profiles
  - Supported by AppleCare
- NoMAD
  - Uses a partner application
  - Offers additional features that are customizable
  - Open Source - Free as in Beer - Community support



# On-Premises Directory - Alternatives



“Your password is...”

“My password is...”

# On-Premises Directory - Alternatives



“Your password is...”



“I’m gonna make my password be...”

# On-Premises Directory - Alternatives



“Your password is...”

“My password was...”



# On-Premises Directory - Alternatives



- Kerberos Tickets
- Mount file shares
- Home directory
- Ongoing password sync

# On-Premises Directory - Alternatives



- Make user account with Setup Assistant
  - “MDM managed user”

# On-Premises Directory - Alternatives



- Make user account with Setup Assistant
  - “MDM managed user”
- Make users with NoLoAD
- Make users with MDM or terminal
  - No user level config profiles



# On-Premises Directory - Alternatives



- Make user account with Setup Assistant
  - “MDM managed user”
- Make users with NoLoAD
- Make users with MDM or terminal
  - No user level config profiles



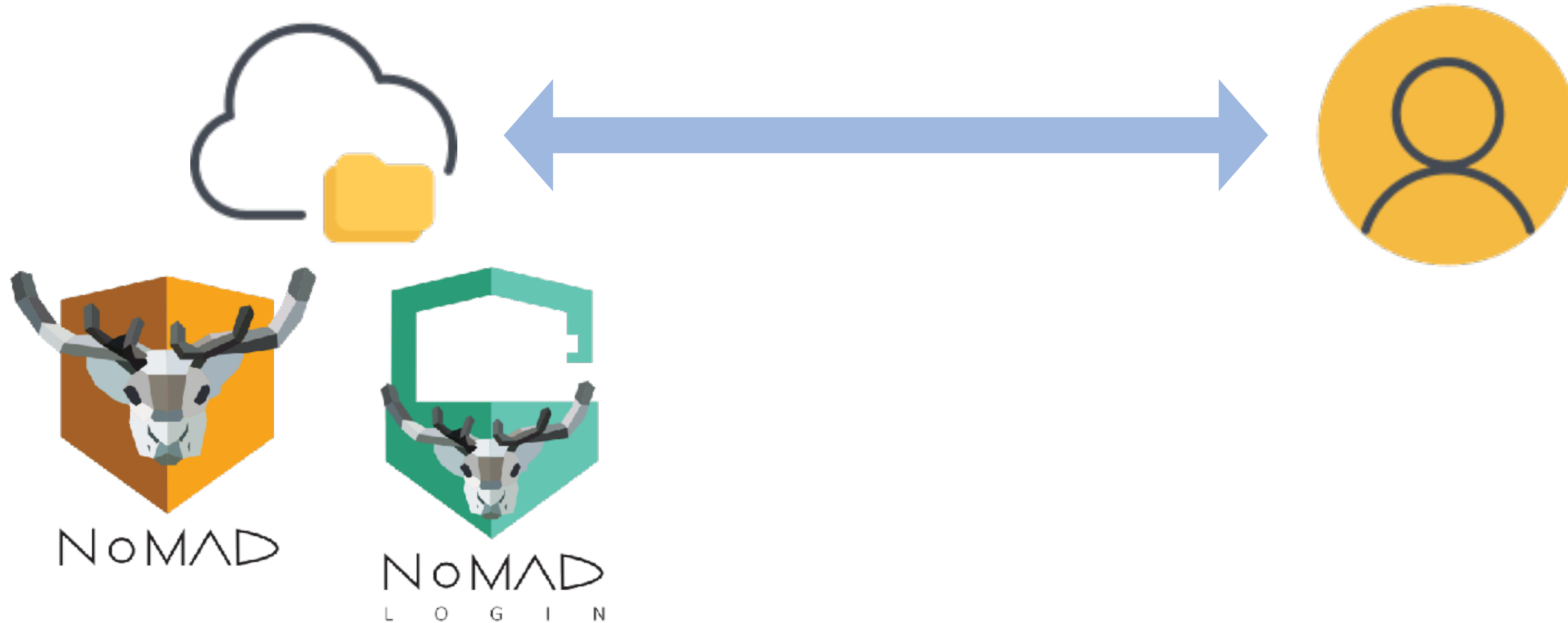
# ~~On-Premises~~ and Cloud Directory Services

# Cloud Directory

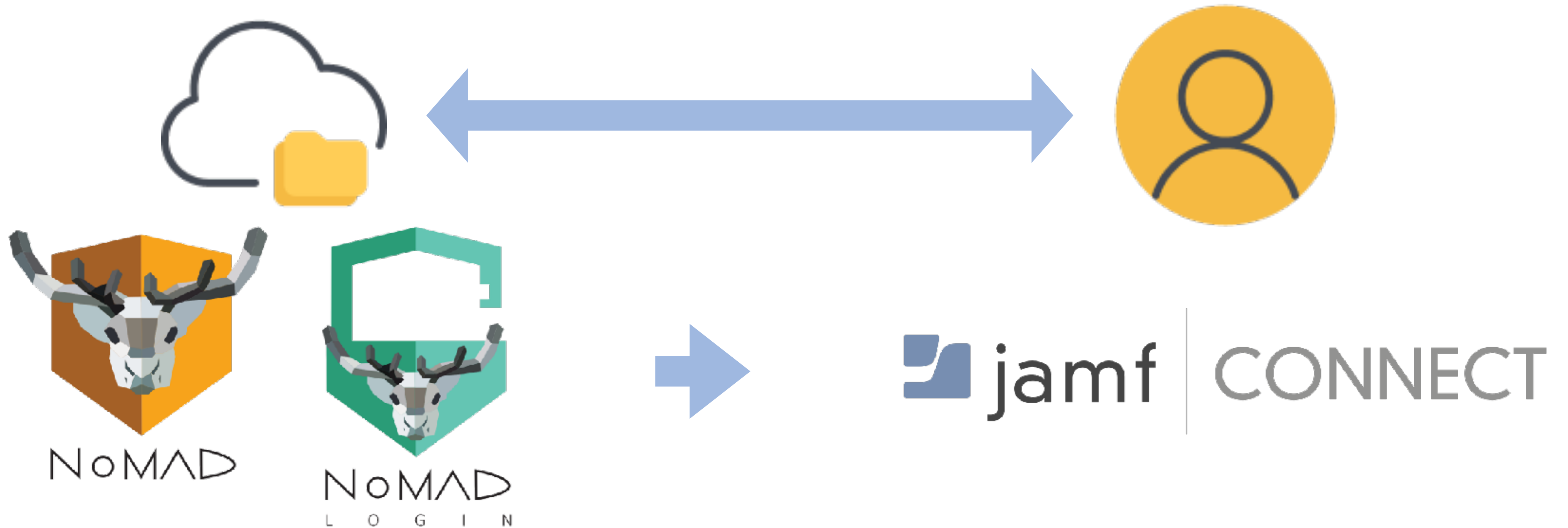




# Cloud Directory



# Cloud Directory



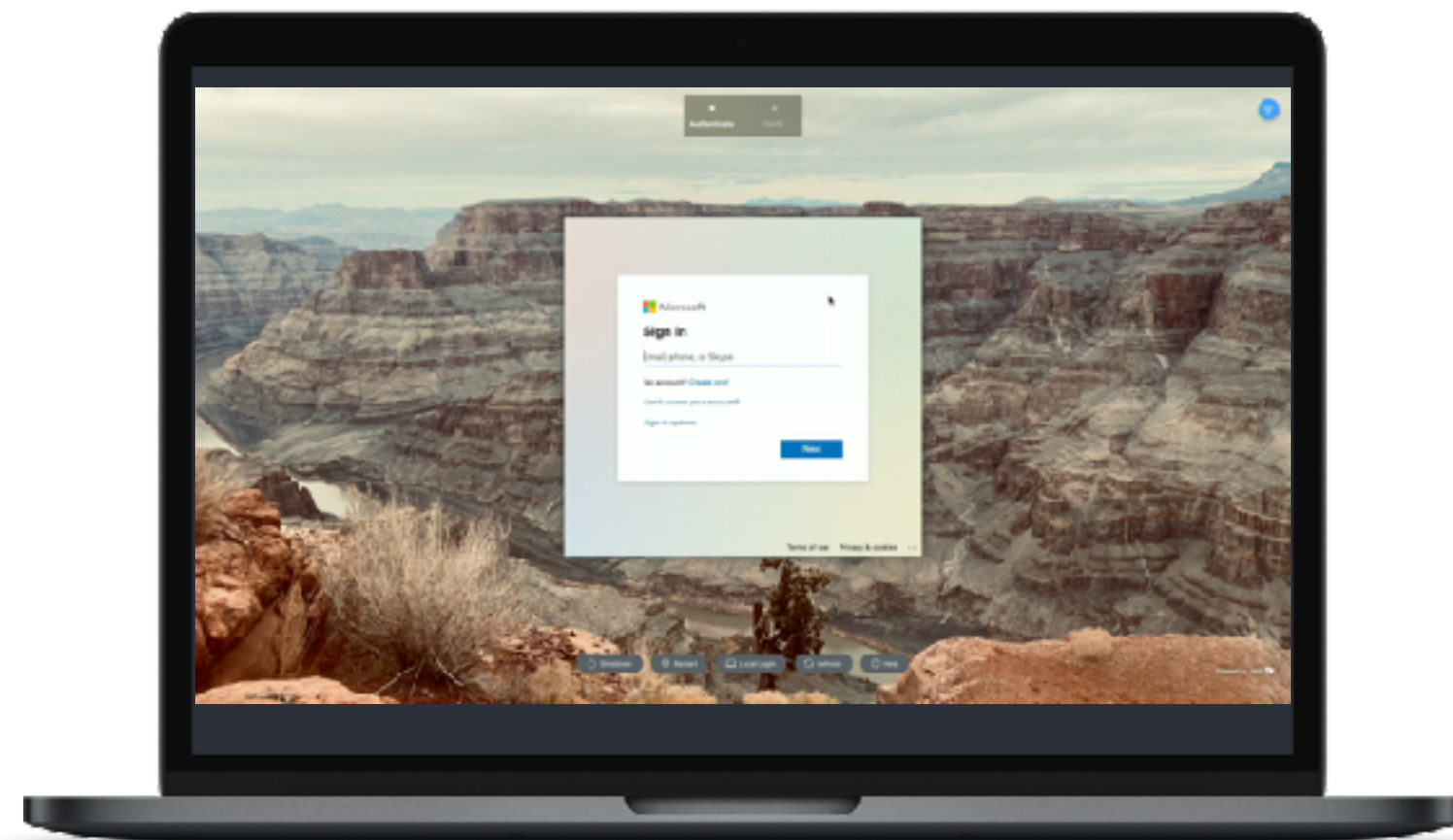
# Cloud Directory



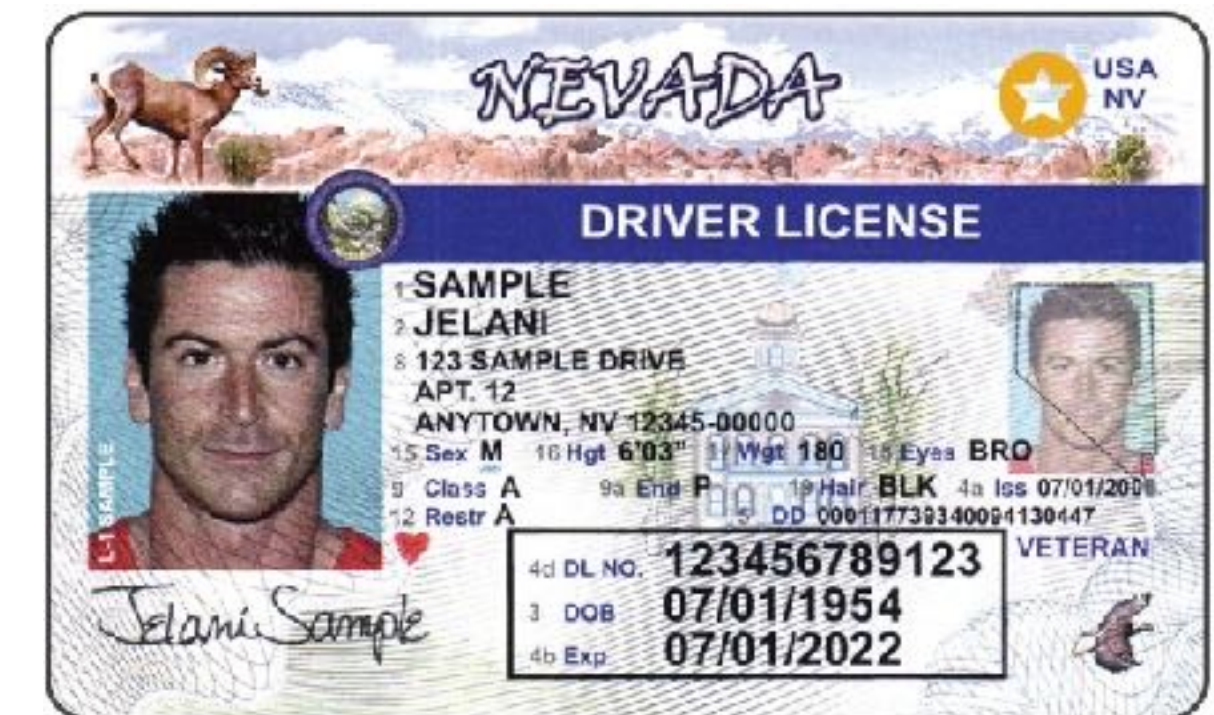
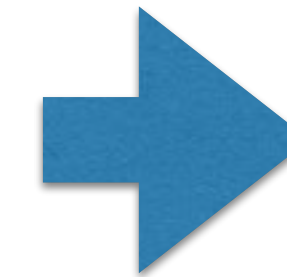
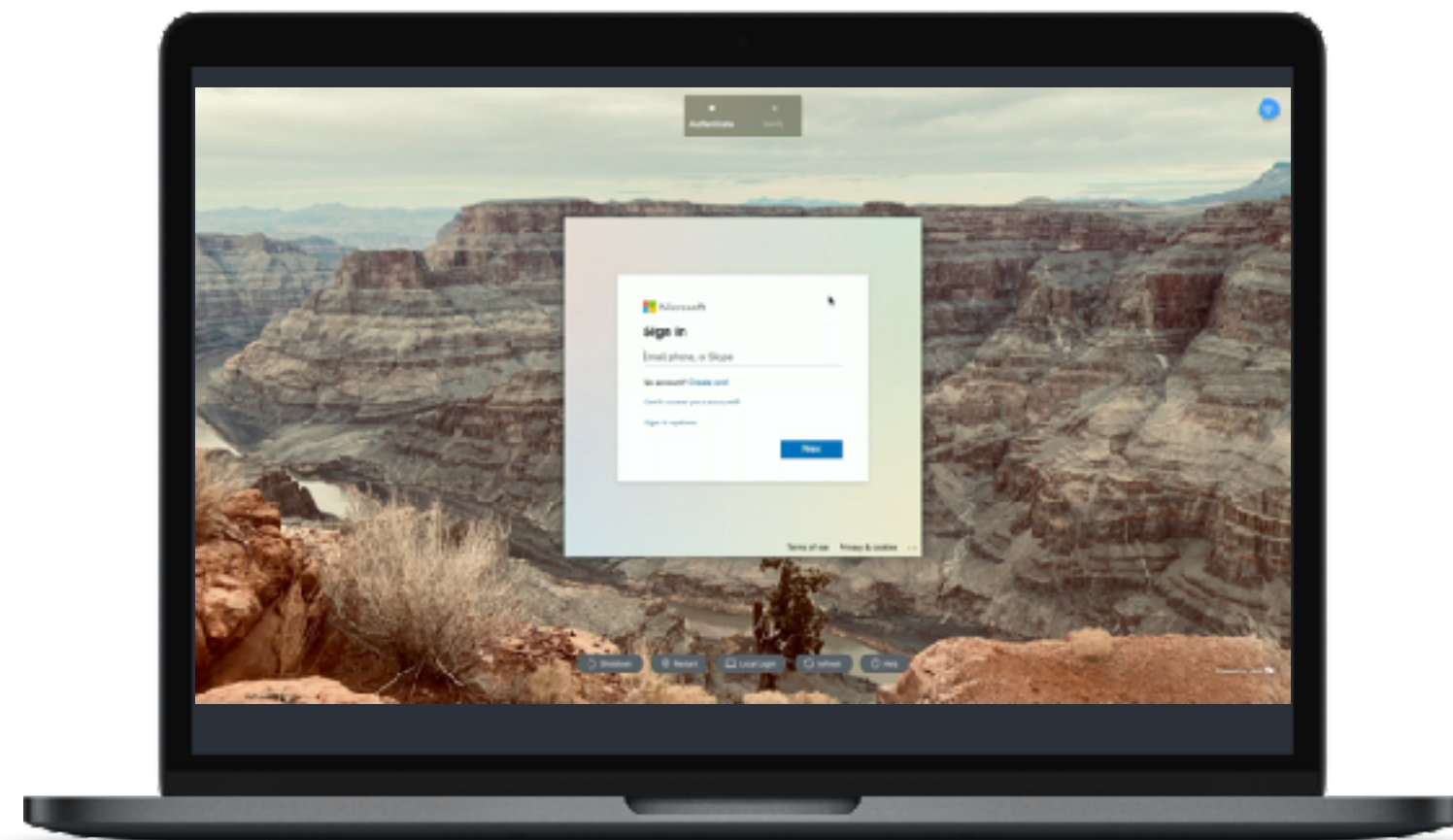
- Jamf Connect
- XCreds
- Mosyle Auth
- Kandji Passport



# Cloud Directory

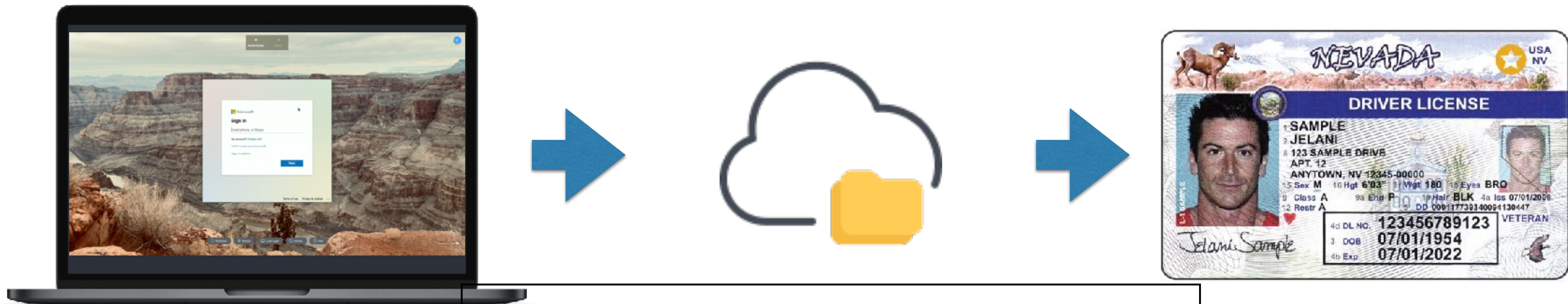


# Cloud Directory





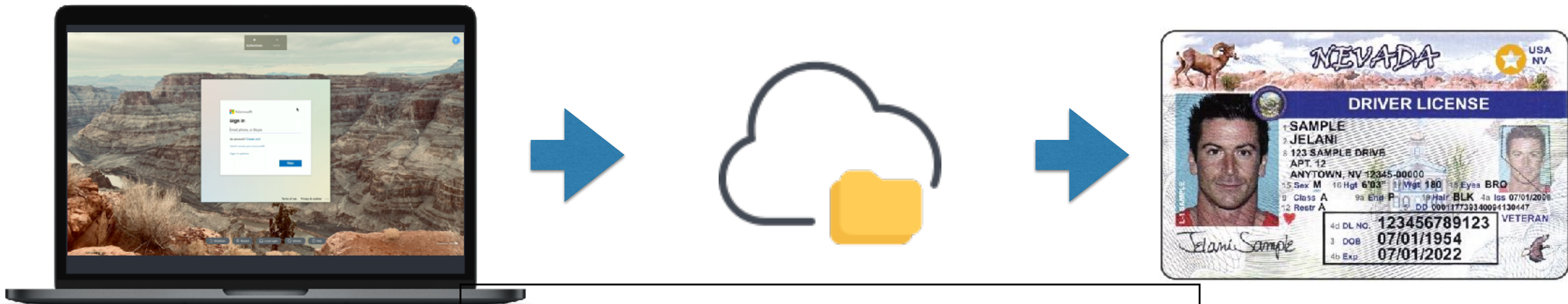
# Cloud Directory



```
<saml2:AttributeStatement xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml2:Attribute Name="http://schemas.xmlsoap.org/claims/Group"
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
    <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="xs:string">macadmin</saml2:AttributeValue>
    <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="xs:string">System Engineers</saml2:AttributeValue>
  </saml2:Attribute>
  <saml2:Attribute Name="RealName"
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
    <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="xs:string">Sean Rabbitt</saml2:AttributeValue>
  </saml2:Attribute>
  <saml2:Attribute Name="UserName"
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
    <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="xs:string">sean.rabbitt</saml2:AttributeValue>
  </saml2:Attribute>
  <saml2:Attribute Name="ShoeSize"
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
    <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="xs:string">10.5 Wide</saml2:AttributeValue>
  </saml2:Attribute>
</saml2:AttributeStatement>
```



# Cloud Directory



Groups

```
<saml2:AttributeStatement xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml2:Attribute Name="http://schemas.xmlsoap.org/claims/Group"
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
    <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="xs:string">macadmin</saml2:AttributeValue>
    <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="xs:string">System Engineers</saml2:AttributeValue>
    </saml2:Attribute>
    <saml2:Attribute Name="RealName"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="xs:string">Sean Rabbitt</saml2:AttributeValue>
      </saml2:Attribute>
      <saml2:Attribute Name="UserName"
        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:type="xs:string">sean.rabbitt</saml2:AttributeValue>
        </saml2:Attribute>
        <saml2:Attribute Name="ShoeSize"
          NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
          <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="xs:string">10.5 Wide</saml2:AttributeValue>
          </saml2:Attribute>
        </saml2:AttributeStatement>
```

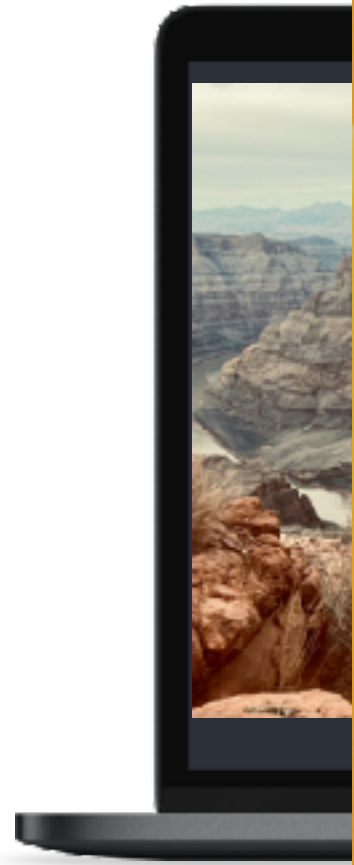
User's "real name"

A user name

\* SAML token stunt double used



Cloud



jamf

Search

Sound

Focus

Screen Time

General

Appearance

Accessibility

Control Center

Siri & Spotlight

Privacy & Security

Desktop & Documents

Displays

Wallpaper

Screen Saver

Energy Saver

Lock Screen

Login Password

Trade Show Standard

Guest User

Add Account...

New Account

Full Name

Account Name:

Password:

Verify:

Password Hint:  
(Recommended)

?

Administrator

✓ Standard

Sharing Only

Group

This will be used as the name for your home folder.

Required

Verify

Hint (Recommended)

Cancel

Create User

Account double used

# Cloud Directory

**But wait, didn't you  
forget something?**



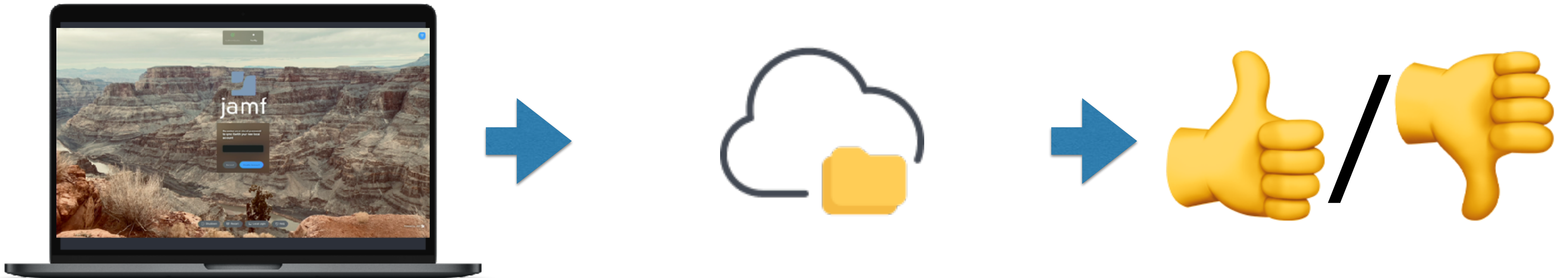
# Cloud Directory



# Cloud Directory



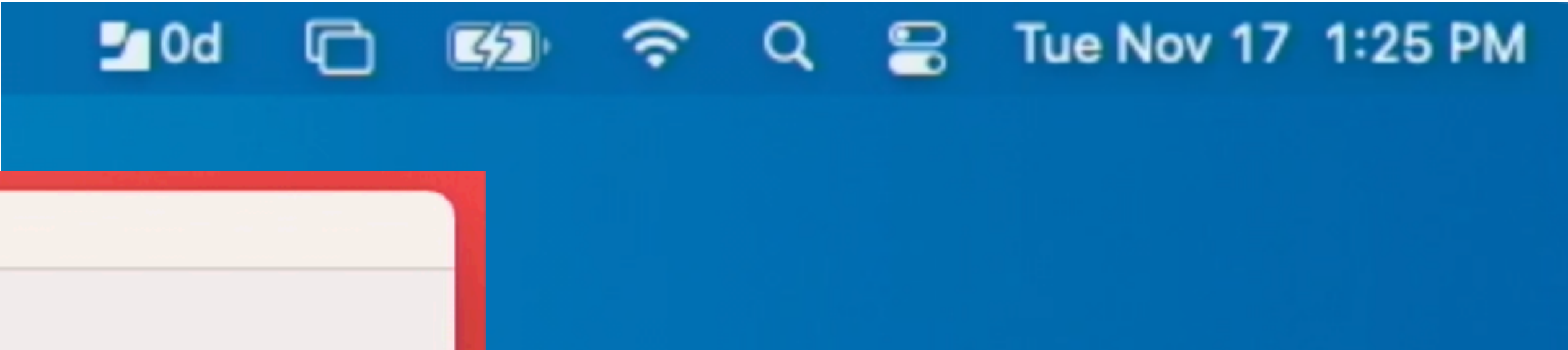
# Cloud Directory



Resource Owner Password Grant  
or  
ROPG



# Cloud Directory



Sign In

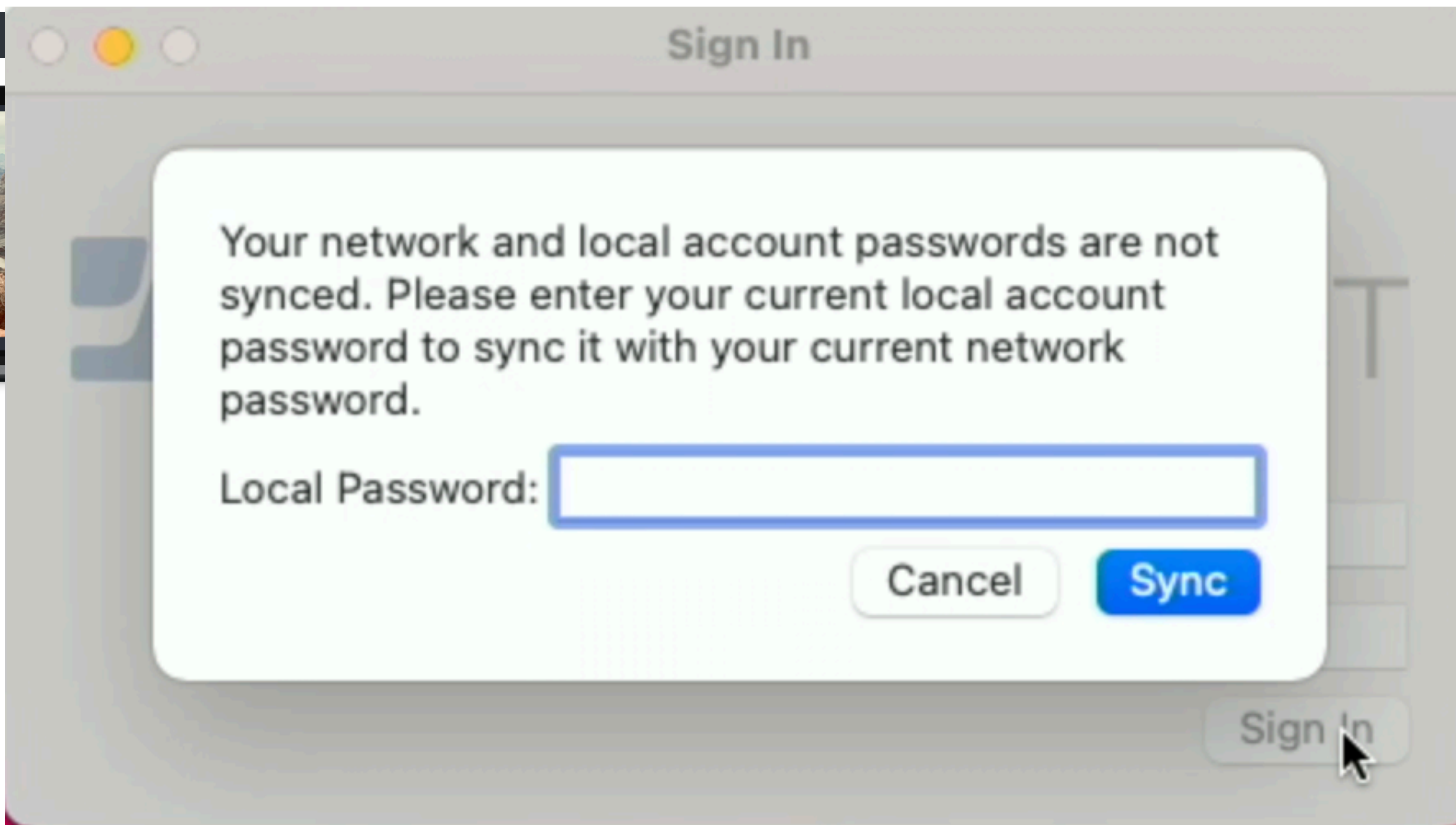
 jamf | CONNECT

Username:

Password:

Sign In

C



# Cloud Directory

- Local account with a “password nag”
  - FileVault and Keychain password kept in sync
  - Grab Kerberos tickets without a bind
  - Mount file shares, home directories, etc.
- Login window could...
  - Force network login
  - Force network login unless no network found
  - Allow or default to local logins





## Sign in

admin.connect@jamfse.io|

No account? [Create one!](#)

[Can't access your account?](#)

Next



Sign-in options

[Terms of use](#) [Privacy & cookies](#) ...



Shut Down



Restart



Local Login



Refresh



Help





## Sign in

admin.connect@jamfse.io|

No account? [Create one!](#)

[Can't access your account?](#)

Next



Sign-in options

[Terms of use](#) [Privacy & cookies](#) ...



Shut Down



Restart



Local Login

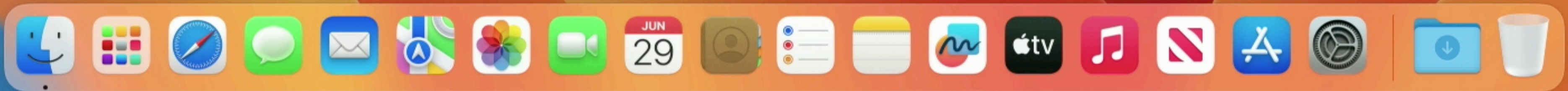


Refresh

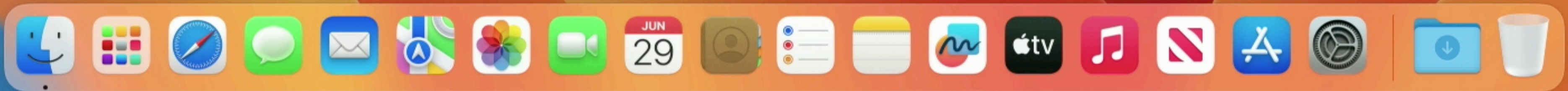


Help









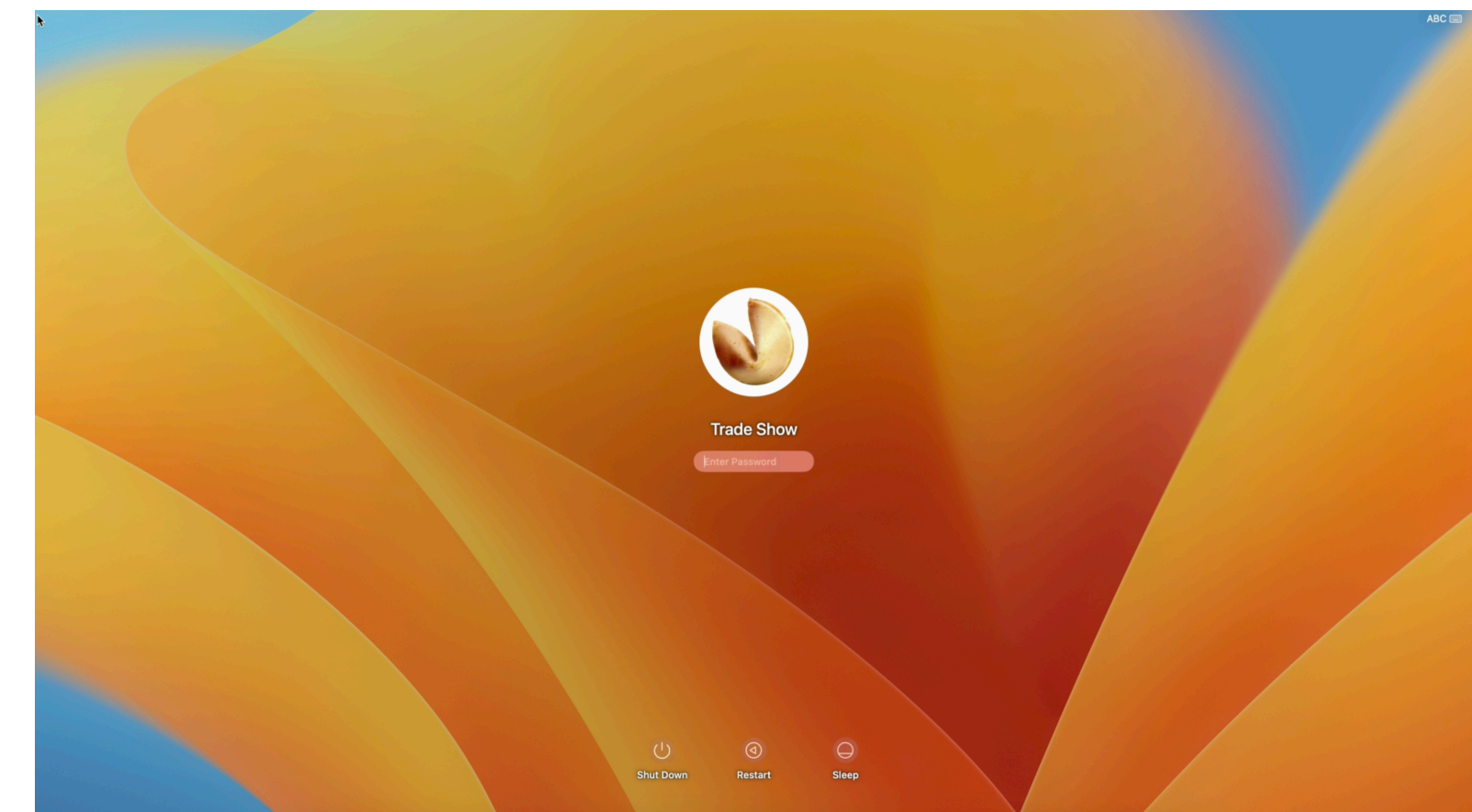
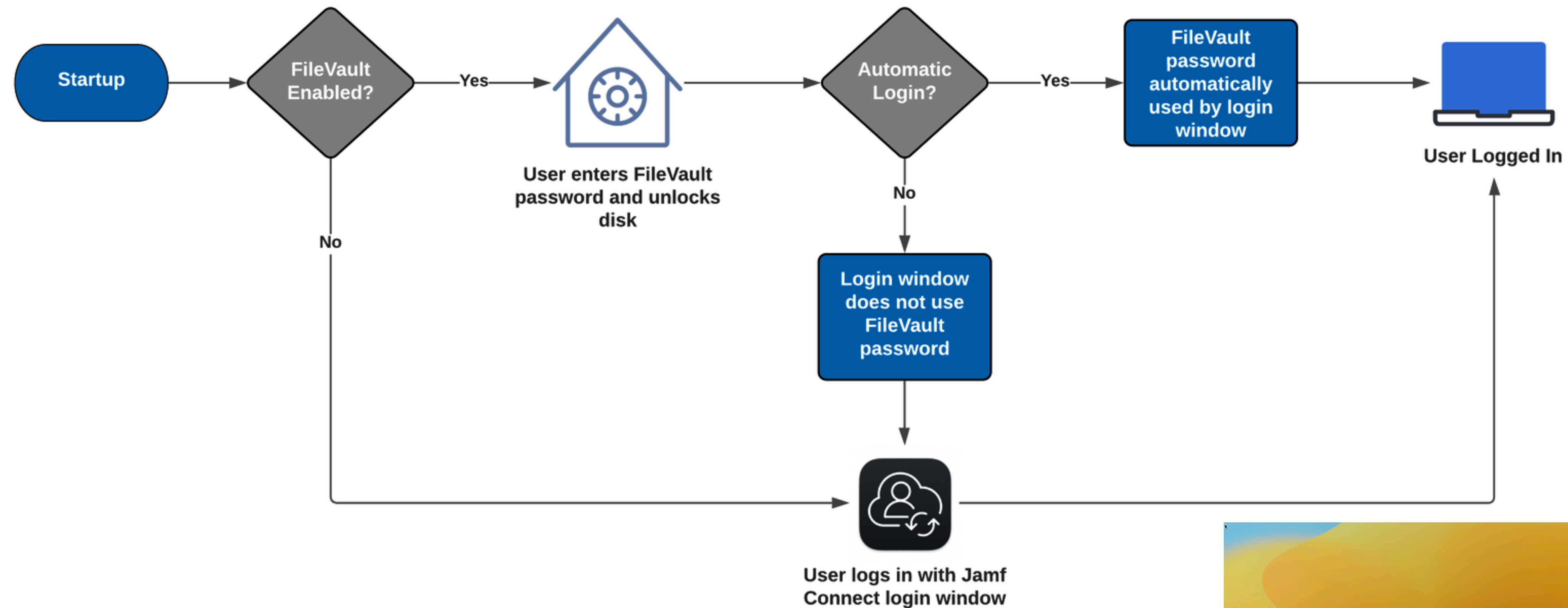


# Cloud Directory

## Sample Uses

- Login and Menu Bar
- Just Login
  - Kiosk / Lab / Shared Desk
  - Ephemeral accounts, deleted after X time
- Just Menu Bar
  - Ongoing password sync, 1:1 machines

# Cloud Directory and FileVault, or “war never changes”





# Cloud identity providers and why those terminal commands are still important

```

52 #look for users created in the last X minutes
53 userAge=60
54
55 # Touch file with list of users to be deleted
56 DELETE_USER_TOUCH_FILE="/Library/Application Support/JAMF/Receipts/.userCleanup"
57 # Credit: Steve Wood
58
59 # Location of the Jamf binary
60 JAMF_BINARY="/usr/local/bin/jamf"
61
62 # Declare list of users variable
63 listOfUsers=""
64
65 # Warn users of what is going to happen
66 responseCode=$(/Library/Application\ Support/JAMF/bin/jamfHelper.app/Contents/MacOS/jamfHelper\
67     -heading "WARNING - THIS APPLICATION CAN DELETE USER DATA" \
68     -cancelButton 1\
69     -button2 "Continue"\
70     -button1 "ABORT"\
71     -windowType utility\
72     -description "This application will search for user accounts created in the last $userAge minutes.  It will mark those accounts for deletion which
73     -title "Jamf Connect Cleanup Script"\
74     -icon "/System/Library/CoreServices/CoreTypes.bundle/Contents/Resources/ToolbarDeleteIcon.icns")
75
76 # If a user hits the abort button, get out of the script and declare an exit code
77 # of 999.  Policy will show as a failure in Jamf Pro logs.
78 if [[ $responseCode = 0 ]]; then
79     exit 999;
80 fi
81
82 # Convert userAge to seconds
83 userAge=$((userAge * 60))
84
85 # For all users who have a password on this machine (eliminates service accounts
86 # but includes the _mbsetupuser and Jamf management accounts...)
87 for user in $(/usr/bin/dscl . list /Users Password | /usr/bin/awk '$2 != "*" {print $1}'); do
88     # If a user has the attribute "OIDCProvider" in their user record, they are
89     # a Jamf Connect user.
90     MIGRATESTATUS=$(/usr/bin/dscl . -read /Users/$user | grep "OIDCProvider: " | /usr/bin/awk '{print $2}')
91     # If we didn't get a result, the variable is empty.  Thus that user is not
92     # a Jamf Connect Login user.
93     if [[ -z $MIGRATESTATUS ]];
94     then
95         # user is not a jamf connect user
96         echo "$user is Not a Jamf Connect User"
97     else

```

```

52 #look for users created in the last X minutes
53 userAge=60
54
55 # Touch file with list of users to be deleted
56 DELETE_USER_TOUCH_FILE="/Library/Application Support/JAMF/Receipts/.userCleanup"
57 # Credit: Steve Wood
58
59 # Location of the Jamf binary
60 JAMF_BINARY="/usr/local/bin/jamf"
61
62 # Declare list of users variable
63 listOfUsers=""
64
65 # Warn users of what is going to happen
66 responseCode=$(/Library/Application\ Support/JAMF/bin/jamfHelper.app/Contents/MacOS/jamfHelper\

```

```

# For all users who have a password on this machine (eliminates service accounts
# but includes the _mbsetupuser and Jamf management accounts...)
for user in $(/usr/bin/dscl . list /Users Password | /usr/bin/awk '$2 != "*" {print $1}'); do
    # If a user has the attribute "OIDCProvider" in their user record, they are
    # a Jamf Connect user.
    MIGRATESTATUS=$(/usr/bin/dscl . -read /Users/$user | grep "OIDCProvider: " | /usr/bin/awk {'print $2'})
    # If we didn't get a result, the variable is empty. Thus that user is not
    # a Jamf Connect Login user.

```

```

84
85 # For all users who have a password on this machine (eliminates service accounts
86 # but includes the _mbsetupuser and Jamf management accounts...)
87 for user in $(/usr/bin/dscl . list /Users Password | /usr/bin/awk '$2 != "*" {print $1}'); do
88     # If a user has the attribute "OIDCProvider" in their user record, they are
89     # a Jamf Connect user.
90     MIGRATESTATUS=$(/usr/bin/dscl . -read /Users/$user | grep "OIDCProvider: " | /usr/bin/awk {'print $2'})
91     # If we didn't get a result, the variable is empty. Thus that user is not
92     # a Jamf Connect Login user.
93     if [[ -z $MIGRATESTATUS ]];
94     then
95         # user is not a jamf connect user
96         echo "$user is Not a Jamf Connect User"
97     else

```



```

127         -description "No local user accounts were created with Jamf Connect Login in the last $userAge seconds. User account may need to be deleted"
128         -title "Jamf Connect Cleanup Script" \
129         -icon "/System/Library/CoreServices/CoreTypes.bundle/Contents/Resources/ProblemReport.icns"
130     else
131         # Otherwise, we found someone - time to tell the user that it's
132         # curtains... lacy, wafting curtains for that user.
133     ###
134     ### YOU CAN EDIT THIS WARNING MESSAGE TO LOCALIZE FOR YOUR IT TEAM HERE
135     ###
136     warningMessage="The following accounts will be deleted within 15 minutes of this policy running:
137
138     $listOfUsers

```

```

167 # Write the list of doomed users to the doomed user file.
168 echo "$listOfUsers" > "$DELETE_USER_TOUCH_FILE"
169
170 # Run a recon so we update the extension attribute
171 # and alert Jamf Pro that this list exists
172 $JAMF_BINARY recon

```

```

162         -description "If you change your mind, delete the file located at $DELETE_USER_TOUCH_FILE immediately." \
163         -title "Jamf Connect Cleanup Script" \
164         -icon "/System/Library/CoreServices/CoreTypes.bundle/Contents/Resources/AlertStopIcon.icns"
165     fi
166
167 # Write the list of doomed users to the doomed user file.
168 echo "$listOfUsers" > "$DELETE_USER_TOUCH_FILE"
169
170 # Run a recon so we update the extension attribute
171 # and alert Jamf Pro that this list exists
172 $JAMF_BINARY recon

```

```
46 # Location of user deadpool list
47 DELETE_USER_TOUCH_FILE="/Library/Application\ Support/JAMF/Receipts/.userCleanup"
48
49 if [ -f "$DELETE_USER_TOUCH_FILE" ]; then
50     echo "<result>TRUE</result>"
51 else
52     echo "<result>FALSE</result>"
53 fi
```

Computers : Smart Computer Groups

← Jamf Connect - User deadpool file exists

Computer Group	Criteria	Reports			
AND/OR	CRITERIA	OPERATOR	VALUE		
▼	Jamf Connect: Does a user deadpool file exist	is ▼	TRUE	▼	Delete

```

67 # SEE NOTES ABOVE - If you want to check for only one admin, set to "1"
68 # If you don't care if there's only a single admin and this script may
69 # fail OR if your environment simply uses all admin accounts anyway, set to "0"
70
71 checkForOnlyOneAdmin=1
72
73 # Location of user deadpool list
74 DELETE_USER_TOUCH_FILE="/Library/Application Support/JAMF/Receipts/.userCleanup"
75 # Credit: Steve Wood
76
77 # Location of the user deadpool list after running script (confirmation file
78 # for auditing)
79 CONFIRM_USER_TOUCH_FILE="/private/tmp/.userDeleted"
80
81 # Location of the Jamf binary
82 JAMF_BINARY=$( which jamf )
83
84 # Convert the space separated list of users into an array for looping through
85 listOfUsers=$(cat "$DELETE_USER_TOUCH_FILE")
86 arrayOfUsers=(${listOfUsers})
87
88 # If we're sanity checking for the "one admin" scenario, look for if there
89 # is only one admin with a securetoken. If true, find any standard account
90 # with a securetoken and mark them for elevation.
91
92 if [[ "$checkForOnlyOneAdmin" -eq 1 ]]; then
93     adminUserCount=0
94     # For all users who have a password on this machine (eliminates service accounts
95     # but includes the _mbsetupuser and Jamf management accounts...)
96     for user in $(/usr/bin/dscl . list /Users Password | /usr/bin/awk '$2 != "*" {print $1}'); do
97         # Is the user an admin
98         isUserAdmin=$(/usr/sbin/dseditgroup -m "$user" -o checkmember admin | /usr/bin/awk {'print $1'})
99         if [ "$isUserAdmin" = "yes" ]; then
100             # Check for securetoken status
101             secureTokenStatus=$(/usr/bin/dscl . -read /Users/"$user" AuthenticationAuthority | /usr/bin/grep -o "SecureToken")
102             # If the account has a SecureToken, increase the securetoken counter
103             if [ "$secureTokenStatus" = "SecureToken" ]; then
104                 ((adminUserCount++))
105             fi
106         fi
107     done
108
109     # If our admin count is less than or equal to 1 (which daymn, if we're less
110     # than one admin account on the box, we've got serious issues and shouldn't
111     # even be here today...) OR if the number of users with a securetoken is
112     # equal to the size of the array of users to be deleted...

```



```

67 # SEE NOTES ABOVE - If you want to check for only one admin, set to "1"
68 # If you don't care if there's only a single admin and this script may
69 # fail OR if your environment simply uses all admin accounts anyway, set to "0"
70
71 checkForOnlyOneAdmin=1
72
73 # Location of user deadpool list
74 DELETE_USER_TOUCH_FILE="/Library/Application Support/JAMF/Receipts/.userCleanup"
75 # Credit: Steve Wood
76
77 # Location of the user deadpool list after running script (confirmation file
78 # for auditing)
79 CONFIRM_USER_TOUCH_FILE="/private/tmp/.userDeleted"
80
81 # Location of the Jamf binary
82 JAMF_BINARY=$( which jamf )

```

# Elevate our eligible account.

echo "Elevating \$elevateThisUser"

/usr/sbin/dseditgroup -o edit -a "\$elevateThisUser" -t user admin

```

96 for user in $(/usr/bin/dscl . list /Users Password | /usr/bin/awk '$2 != "*" {print $1}'); do
97     # Is the user an admin
98     isUserAdmin=$(/usr/sbin/dseditgroup -m "$user" -o checkmember admin | /usr/bin/awk {'print $1'})
99     if [ "$isUserAdmin" = "yes" ]; then
100         # Check for securetoken status
101         secureTokenStatus=$(/usr/bin/dscl . -read /Users/"$user" AuthenticationAuthority | /usr/bin/grep -o "SecureToken")
102         # If the account has a SecureToken, increase the securetoken counter
103         if [ "$secureTokenStatus" = "SecureToken" ]; then
104             ((adminUserCount++))
105         fi
106     fi
107 done
108
109 # If our admin count is less than or equal to 1 (which daymn, if we're less
110 # than one admin account on the box, we've got serious issues and shouldn't
111 # even be here today...) OR if the number of users with a securetoken is
112 # equal to the size of the array of users to be deleted...

```

```

158 # For every user in the list, delete the user account with the Jamf binary
159 for user in ${arrayOfUsers[@]}; do
160
161     echo "Deleting $user"
162     #####
163     #####
164     ### HERE'S WHERE YOU UNCOMMENT STUFF FOR DATA LOSS TO PURPOSELY HAPPEN!! ###
165     #####
166     #####
167     # It's not that I don't trust you. I don't trust anyone.
168     echo "$JAMF_BINARY deleteAccount -username $user -deleteHomeDirectory"
169     #$JAMF_BINARY deleteAccount -username "$user" -deleteHomeDirectory
170 done
171
172 # Demote our user back to standard user if needed
173 if [[ -z $elevateThisUser ]]; then
174     echo "We didn't have to elevate a user in this case."
175 else
176     echo "Demoting $elevateThisUser to standard account"
177     /usr/sbin/dseditgroup -o edit -d "$elevateThisUser" -t user admin
178 fi
179
180 # Move the delete file for auditing purposes
181 /bin/mv "$DELETE_USER_TOUCH_FILE" "$CONFIRM_USER_TOUCH_FILE"
182
183 # Run a recon to clear out the extension attribute / smart computer group for
184 # running this process.
185 $JAMF_BINARY recon

```



```

158 # For every user in the list, delete the user account with the Jamf binary
159 for user in ${arrayOfUsers[@]}; do
160
161     echo "Deleting $user"
162     #####
163     #####

```

```

echo "Deleting $user"
#####
#####
### HERE'S WHERE YOU UNCOMMENT STUFF FOR DATA LOSS TO PURPOSELY HAPPEN!! ###
#####
#####
# It's not that I don't trust you. I don't trust anyone.
echo "$JAMF_BINARY deleteAccount -username $user -deleteHomeDirectory"
#$JAMF_BINARY deleteAccount -username "$user" -deleteHomeDirectory

```

```

180 # Move the delete file for auditing purposes
181 /bin/mv "$DELETE_USER_TOUCH_FILE" "$CONFIRM_USER_TOUCH_FILE"
182
183 # Run a recon to clear out the extension attribute / smart computer group for
184 # running this process.
185 $JAMF_BINARY recon

```



**[https://github.com/sean-rabbitt/  
JIT-user-deletion-with-jamf-  
connect](https://github.com/sean-rabbitt/JIT-user-deletion-with-jamf-connect)**



# The Future: Platform Single Sign-On

Or, rampant speculation because ain't nobody has released  
this to the public yet

# Single Sign-On Extension for Enterprise

Computers : Configuration Profiles

← Okta Single Sign-On Extension for macOS

Options

Scope

☐ Show in Jamf Pro Dashboard

Search...

General

Single Sign-On Extensions  
1 payload configured

Single Sign-on Extension

1 payload configured

Configure app extensions that perform single sign-on (macOS 10.15 or later, User Approved MDM required).

SSO

Payload Type

The payload type

SSO

Extension Identifier

Bundle identifier of the app extension that performs single sign-on

com.okta.mobile.auth-service-extension

Team Identifier

The team identifier of the app extension that performs single sign-on

B7F62B65BN

Sign-on Type

Sign-on authorization type

Credential

Realm

Realm name for the Credential-type payload. This value must be properly capitalized.

Okta Device

Hosts

Hostnames that can be authenticated through the app extension. Names must be unique for all configured Single Sign-On Extensions payloads.

jamfse-oie.oktapreview.com



# Single Sign-On Extension for Enterprise

← Microsoft Enterprise Single Sign-On Plug-in

Options

Scope

☐ Show in Jamf Pro Dashboard

Search...

General

Application & Custom Settings1 payload configured

Single Sign-On Extensions1 payload configured

Single Sign-on Extensions

1 payload configured

Single Sign-on Extension

Configure app extensions that perform single sign-on (macOS 10.15 or later, User Approved MDM required).

Payload Type

The payload type

SSO

Extension Identifier

Bundle identifier of the app extension that performs single sign-on

com.microsoft.CompanyPortalMac.ssoextension

Team Identifier

The team identifier of the app extension that performs single sign-on

UBF8T346G9

Sign-on Type

Sign-on authorization type

Redirect

URLs

URLs of identity providers where the app performs single sign-on. The URLs must begin with http:// or https:// and be unique for all configured Single Sign-On Extensions payloads. Query parameters and URL fragments are not allowed.


https://login.microsoftonline.com

https://login.microsoft.com

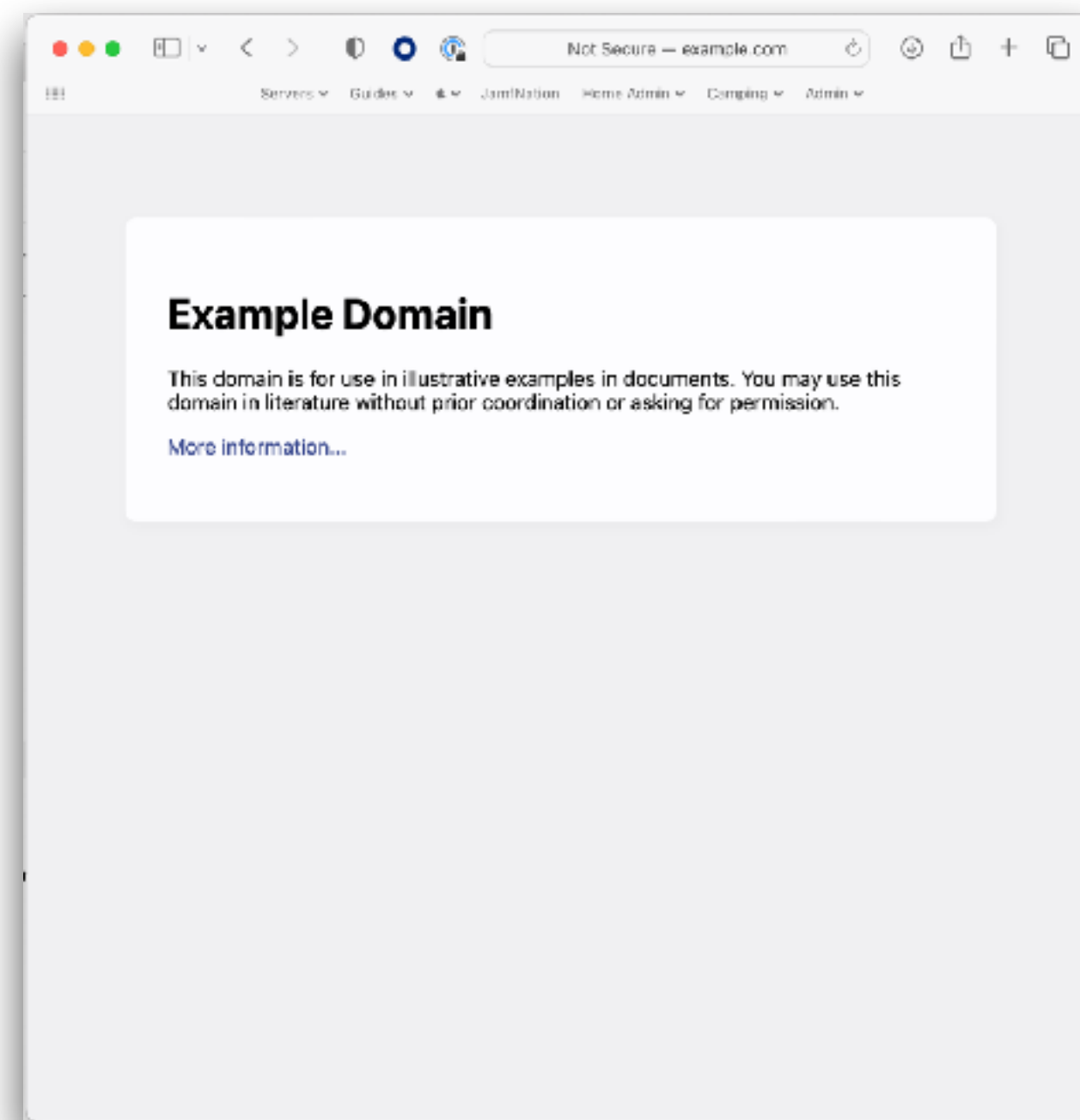
https://sts.windows.net

https://login.partner.microsoftonline.cn

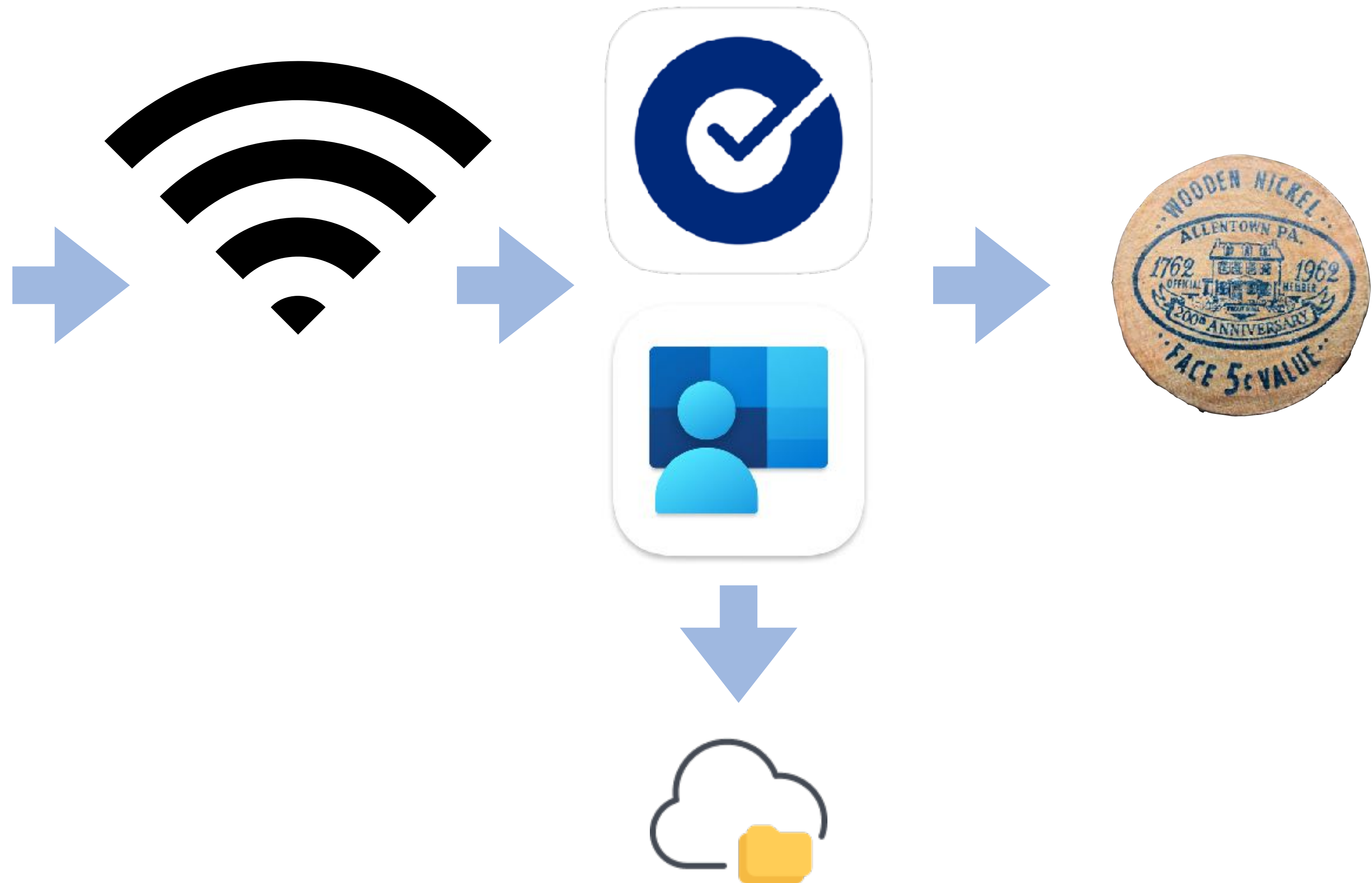
https://login.chinacloudapi.cn

 jamf

# Single Sign-On Extension for Enterprise

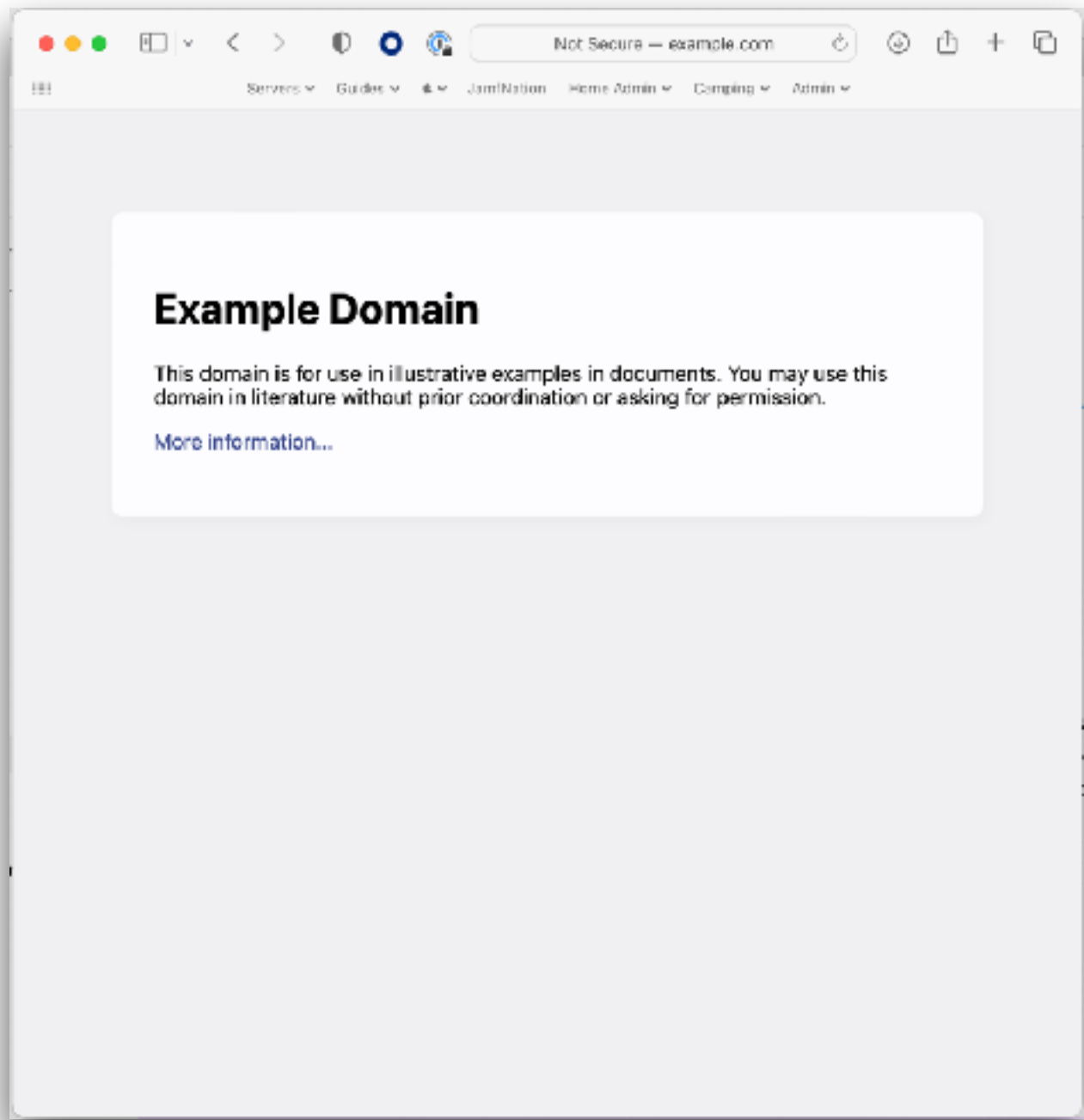


open <https://example.com/login>

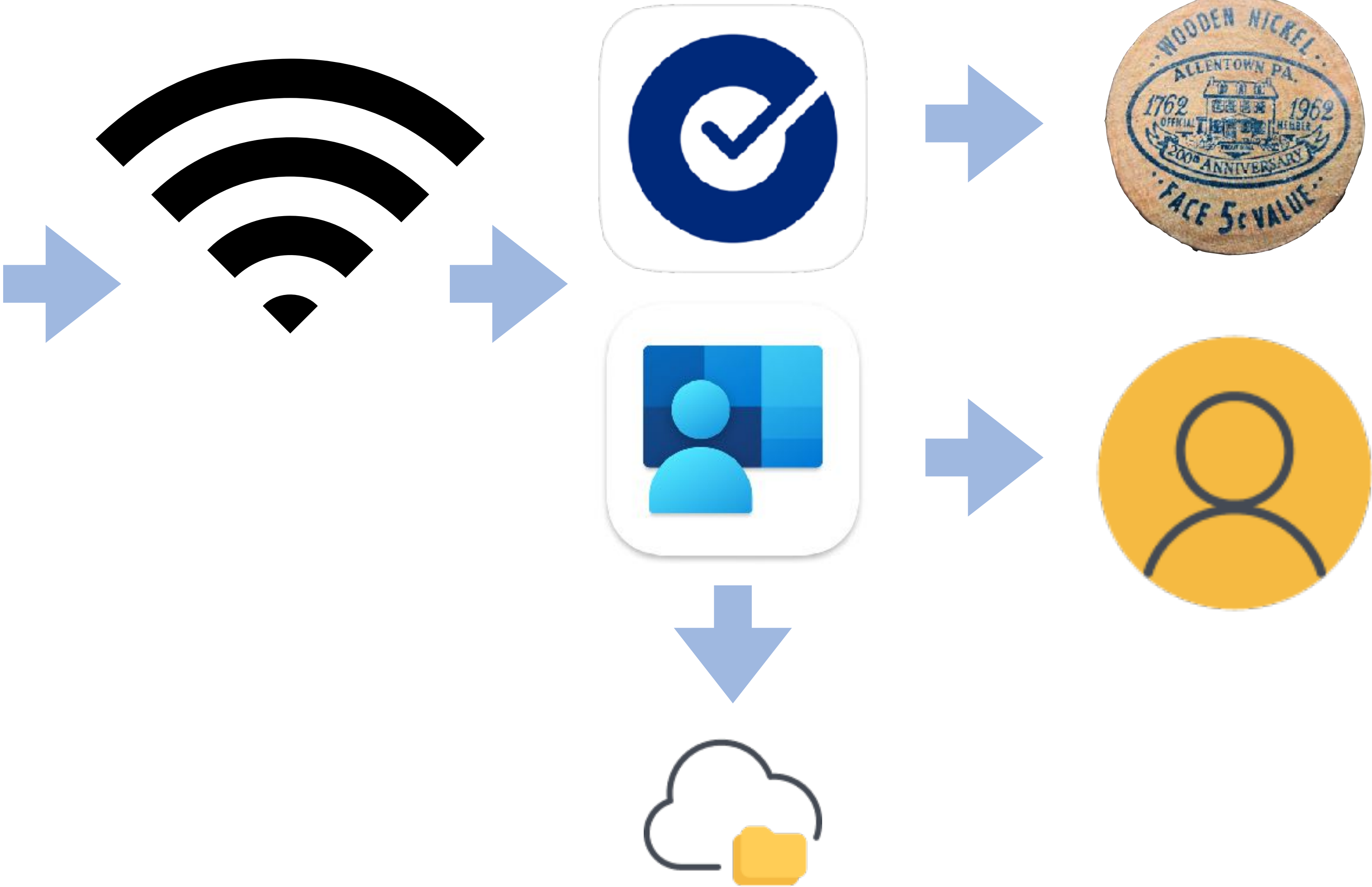




# Platform Single Sign-On Extension



open <https://example.com/login>











# Platform Single Sign On - as of macOS Ventura

Feature	Platform Single Sign On	Jamf Connect & Others
Works at login window		✓
Makes local user account		✓
Admin / Standard rights management		✓
Works in Zero Touch Enrollment flow		✓
Can enforce network only logins		✓
Can enforce MFA for offline auth		✓ (Depends on tool used)
Keeps local account in sync with IdP	✓	✓
Kerberos support	✓ (with Kerberos SSOe)	✓
Automatically logs in to cloud IdP gated apps	✓	
Screensaver Unlock	?	



**We don't talk about  
betas in public  
forums.**

**But it's not really beta...**

**<https://appleseed.apple.com>**

**<https://beta.apple.com/it>**



# Platform Single Sign On - as of macOS Sonoma

Feature	Platform Single Sign On	Jamf Connect & Others
Works at login window	✓	✓
Makes local user account	✓ (After first admin account created)	✓
Admin / Standard rights management	✓	✓
Works in Zero Touch Enrollment flow	🙋	✓
Can enforce network only logins		✓
Can enforce MFA for offline auth		✓ (Jamf Connect only)
Keeps local account in sync with IdP	✓	✓
Kerberos support	✓ (with Kerberos SSOe)	✓
Automatically logs in to cloud IdP gated apps	✓	
Screensaver Unlock	✓	
PIV / SmartCard Support	✓	

# Platform Single Sign On - as of macOS Sonoma

## Authentication Scenarios:

- Password - Local account password sync with the IdP
- Password with WS-Trust - IdP doesn't know password - SAML token auth
- User Secure Enclave Key - Auth to IdP without a password - still local password
- SmartCard - Auth with cert on PIV - local password maybe?



# Platform Single Sign On - as of macOS Sonoma

## Authentication Scenarios:

- Password - Local account password sync with the IdP
- Password with WS-Trust - IdP doesn't know password - SAML token auth
- User Secure Enclave Key - Auth to IdP without a password - still local password
- SmartCard - Auth with cert on PIV - local password maybe?

## Group Membership:

- Pass up to 100 IdP based groups to local macOS device
- Local UNIX group membership determines admin/standard/sudo rights

# Platform Single Sign On - as of macOS Sonoma

## Shared Device Registration





# Platform Single Sign On - as of macOS Sonoma

## Shared Device Registration



## User Registration



**Final thoughts :**  
Local User Accounts  
Network Accounts  
Cloud Identity Accounts  
Platform Single Sign-On



# Final Thoughts

- macOS is UNIX
- FileVault gonna FileVault
- Tying to a directory introduces challenges
- Challenges can be overcome
- Let's see what happens with PSSOe in the future
- macOS is still UNIX

<https://github.com/sean-rabbitt>  
for slides

I'll be at Jamf's booth after this.



A woman with blonde hair is seated at a desk, looking at a tablet. She is holding a smartphone in her left hand. In the background, a man is seated at another desk, looking at a laptop. The image has a blue and purple gradient overlay.

**Thank you.**