



Managing user identity on Macs



Sean Rabbitt

Sr Consulting Engineer,
Identity and Access Mgmt

PRESENTING TO

University of Utah - MacAdmins
October 2023

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 | Login Window Alternatives

macOS has a built-in user session manager called loginwindow. But there's options.

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

macOS is UNIX



macOS is **UNIX**



Local Accounts and Groups

Short Name
Real Name
UID
Primary Group
Home Directory



Hierarchical File Structure

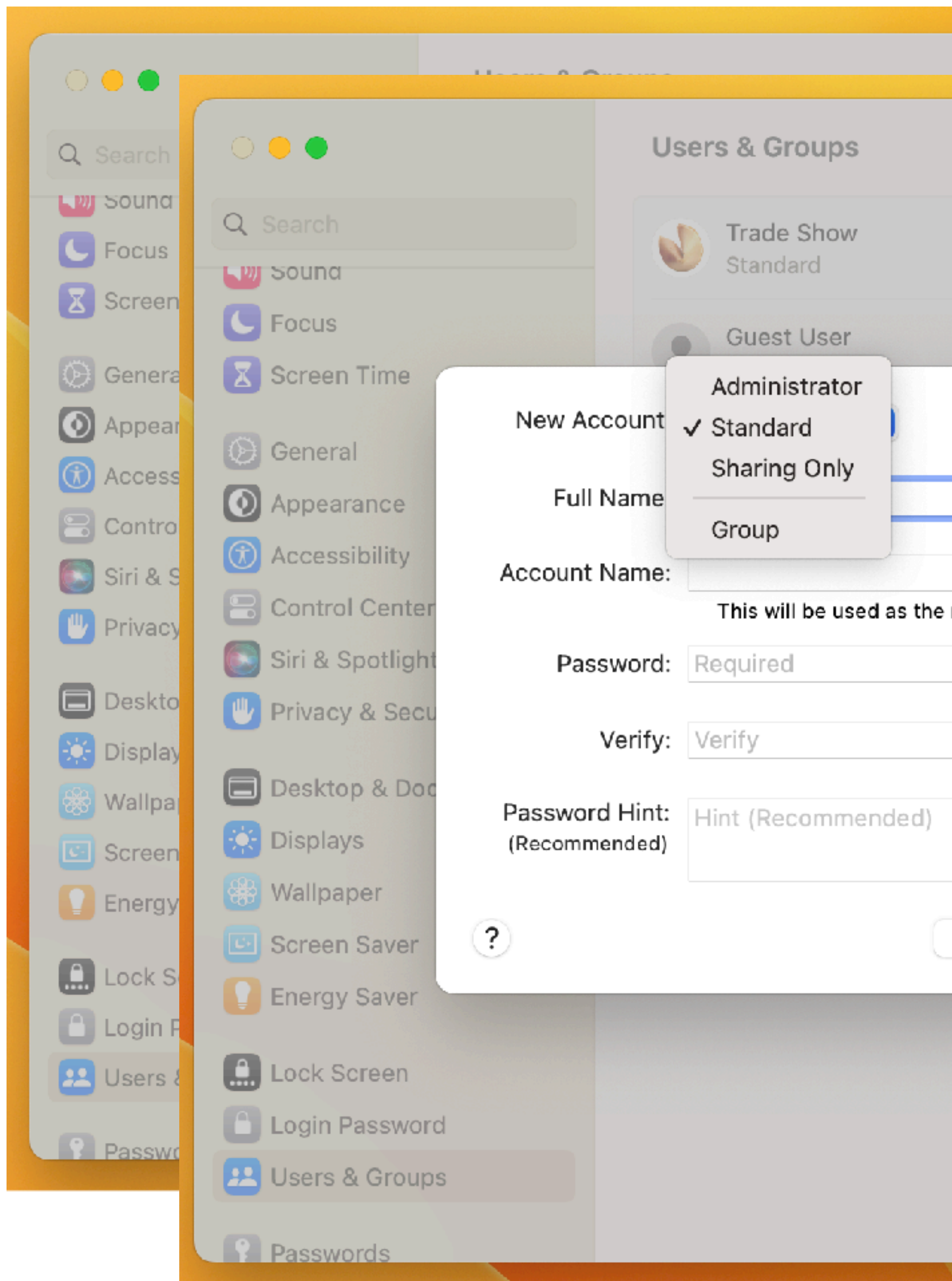
File Owner
Group Owner
Read / Write / Execute
Other Apple Specific Magic




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

User ID

Group

User name

Full name

Login shell

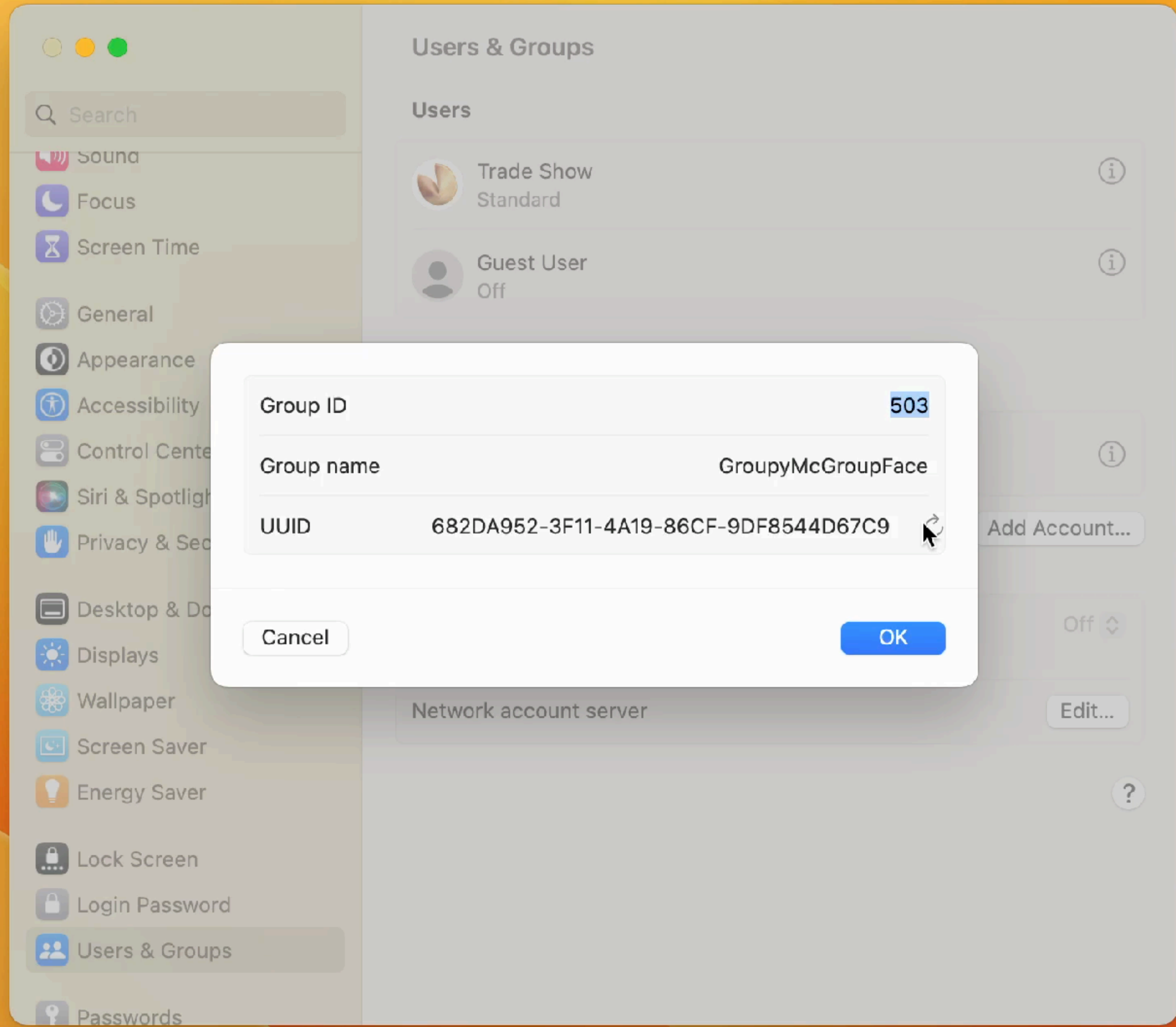
Home directory


Home directory

UUID

Apple ID

Aliases



 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

+ -

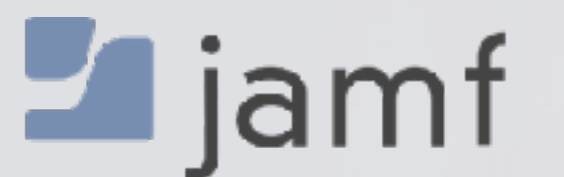
Cancel

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"
```


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
```

dscl

```
dscl . 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
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 _
```

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 -
    FFFFEEEE-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 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?

```
dsc1 . -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
- 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

pwpolicy

Wait, it does more than reset passwords?

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

Disable a local user from logging in

```
pwpolicy -u user -disableuser
```

```
pwpolicy -u user -enableuser
```

Do something terrible and set a local account policy manually

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

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

```
pwpolicy -clearaccountpolicies *
```

Pushing settings via MDM...

Untitled — Edited

Passcode

- Allow simple value**
Permit the use of repeating, ascending, and descending character sequences
- Require alphanumeric value**
Requires passcode to contain at least one letter and one number
- ▾ **Minimum passcode length**
Smallest number of passcode characters allowed
- ▾ **Minimum number of complex characters**
Smallest number of non-alphanumeric characters allowed
- ▢ ▾ **Maximum passcode age (1-730 days, or none)**
Days after which passcode must be changed
- ▢ ▾ **Maximum Auto-Lock**
Longest auto-lock time available to the user
- ▢ ▾ **Passcode history (1-50 passcodes, or none)**
Number of unique passcodes before reuse
- ▢ ▾ **Maximum grace period for device lock**
Longest device lock grace period available to the user
- ▾ **Maximum number of failed attempts**
Number of passcode entry attempts allowed before all data on device will be erased

General Mandatory

Restrictions Not configured

Domains Not configured

Global HTTP Proxy Not configured

DNS Proxy Not configured

Content Filter Not configured

Certificates Not configured

Certificate Transparency Not configured

Passcode 1 Payload Configured

Wi-Fi Not configured

VPN Not configured

AirPlay

- INVENTORY
 - Search Inventory
 - Search Volume Content
 - Licensed Software
- CONTENT MANAGEMENT
 - Policies
 - Configuration Profiles
 - Restricted Software
 - Mac Apps
 - Patch Management
 - eBooks
- GROUPS
 - Smart Computer Groups
 - Static Computer Groups
 - Classes
- ENROLLMENT
 - Enrollment Invitations
 - PreStage Enrollments
- SETTINGS
 - Management Settings

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 Exclude all

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

Setting **Include**

Require Passcode Enforce setting passcode on the computer.	<input type="checkbox"/>
Complex Passcode Passcode cannot contain repeating, ascending, and descending character sequences.	Enforce Ignore <input type="checkbox"/>
Alphanumeric Value Passcode must contain at least one letter and one number.	Enforce Ignore <input type="checkbox"/>
Minimum Passcode Length Smallest number of passcode characters allowed	<input type="text" value="1"/> <input type="checkbox"/>
Minimum Number of Complex Characters Smallest number of non-alphanumeric characters allowed	<input type="text" value="1"/> <input type="checkbox"/>
Maximum Passcode Age Number of days until the passcode must be changed (1-730)	<input type="text" value=""/> <input type="checkbox"/>
Passcode History Number of unique passcodes before reuse (1-50)	<input type="text" value=""/> <input type="checkbox"/>
Maximum Auto-Lock Number of minutes before the computer automatically locks	<input type="text" value=""/> <input type="checkbox"/>
Maximum Grace Period for Computer Lock Period of inactivity before the passcode is required to unlock the computer	<input type="text" value=""/> <input type="checkbox"/>
Maximum Number of Failed Attempts Number of passcode entry attempts allowed before the computer is locked	<input type="text" value="11"/> <input 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 type="text" value=""/> <input type="checkbox"/>
Change at Next Authentication (macOS 10.13 or later) Force password reset on next user authentication.	Enforce Ignore <input type="checkbox"/>

macOS iOS tvOS Search Preferences

Untitled

Login Window

Computers : Configuration Profiles

← Test Login Window Policy

Login Window

Login Window settings

macOS

Available System Domains

Restrictions macOS, iOS, and tvOS

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

Notifications Not configured

Parental Controls Not configured

Passcode Not configured

Printing Not configured

Privacy Preferences Policy Control Not configured

Disable macOS 10

Show

Hidden Us Hides users

Name

List of

Show S

Loginwindow

Window Options Access Pass

- Disable automatic login if FileVault is disabled
- Disable automatic login if FileVault is enabled
- Disable >console login
- Enable external accounts
- Mac computer administrators may refresh content or disable management
- Reopen windows when logging back in
- Set Mac computer name to computer record name

Login Window

Window Options Access Script

- Show password hint when needed and available
- Disable automatic login

Login Window

Window Options Access Script

Allow The users and groups that can login at this computer

NAME	SERVER
+ Add	

Deny The users and groups that cannot login at this computer

USER	SERVER
+ Add	

- Local-only users may log in
- Local-only users use available workgroup settings
- Ignore workgroup nesting
- Combine available workgroup settings
- Always show workgroup dialog during login

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

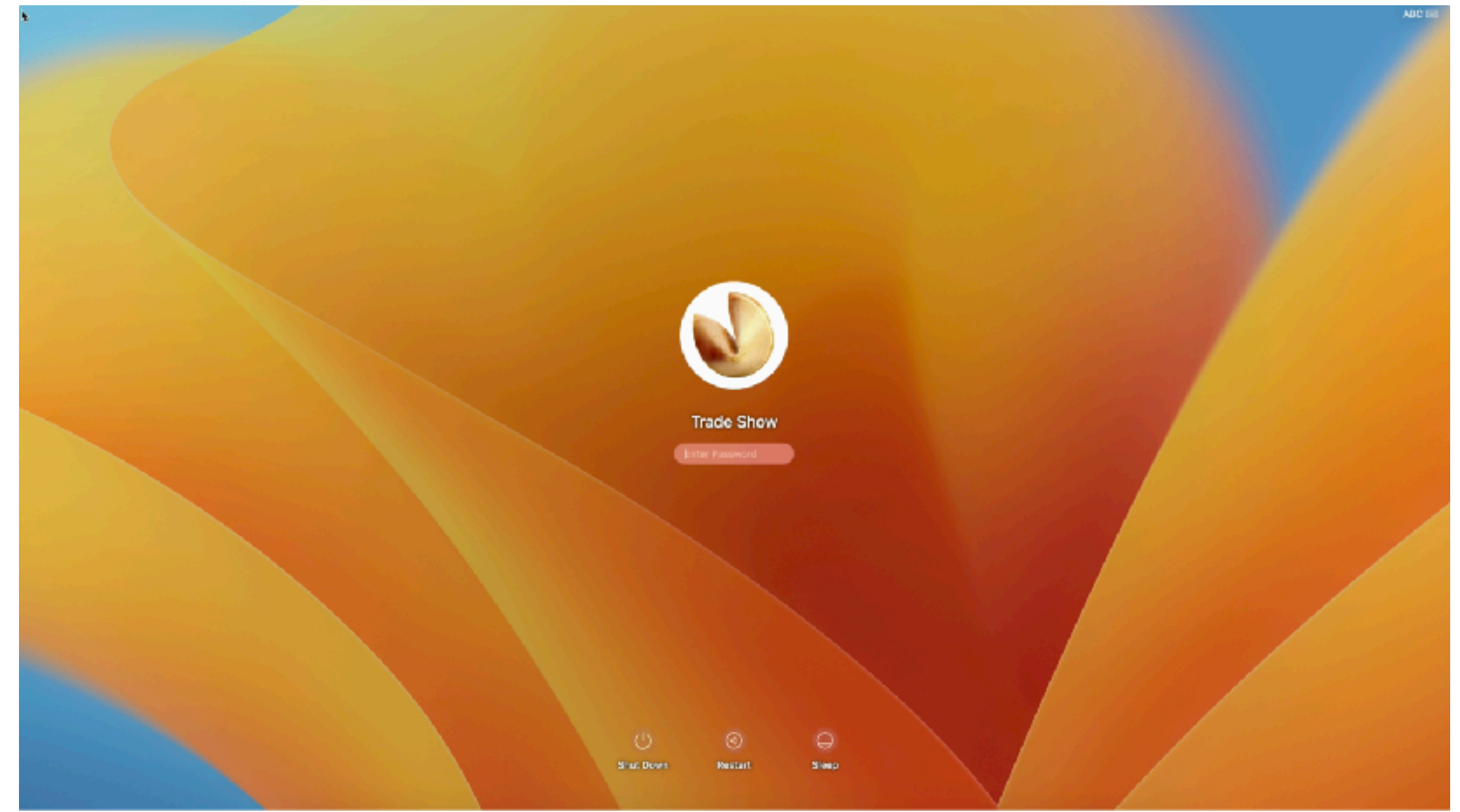
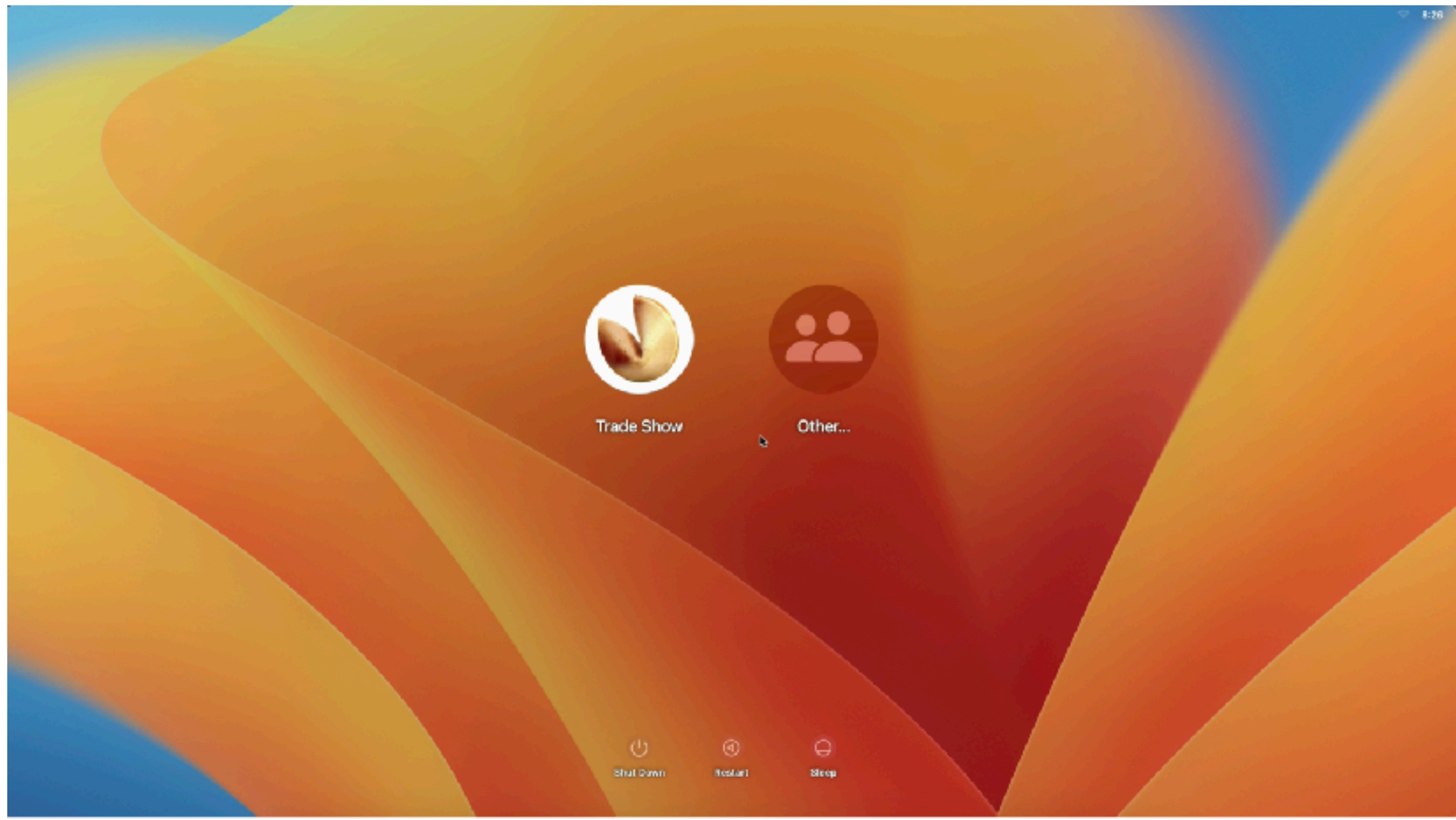
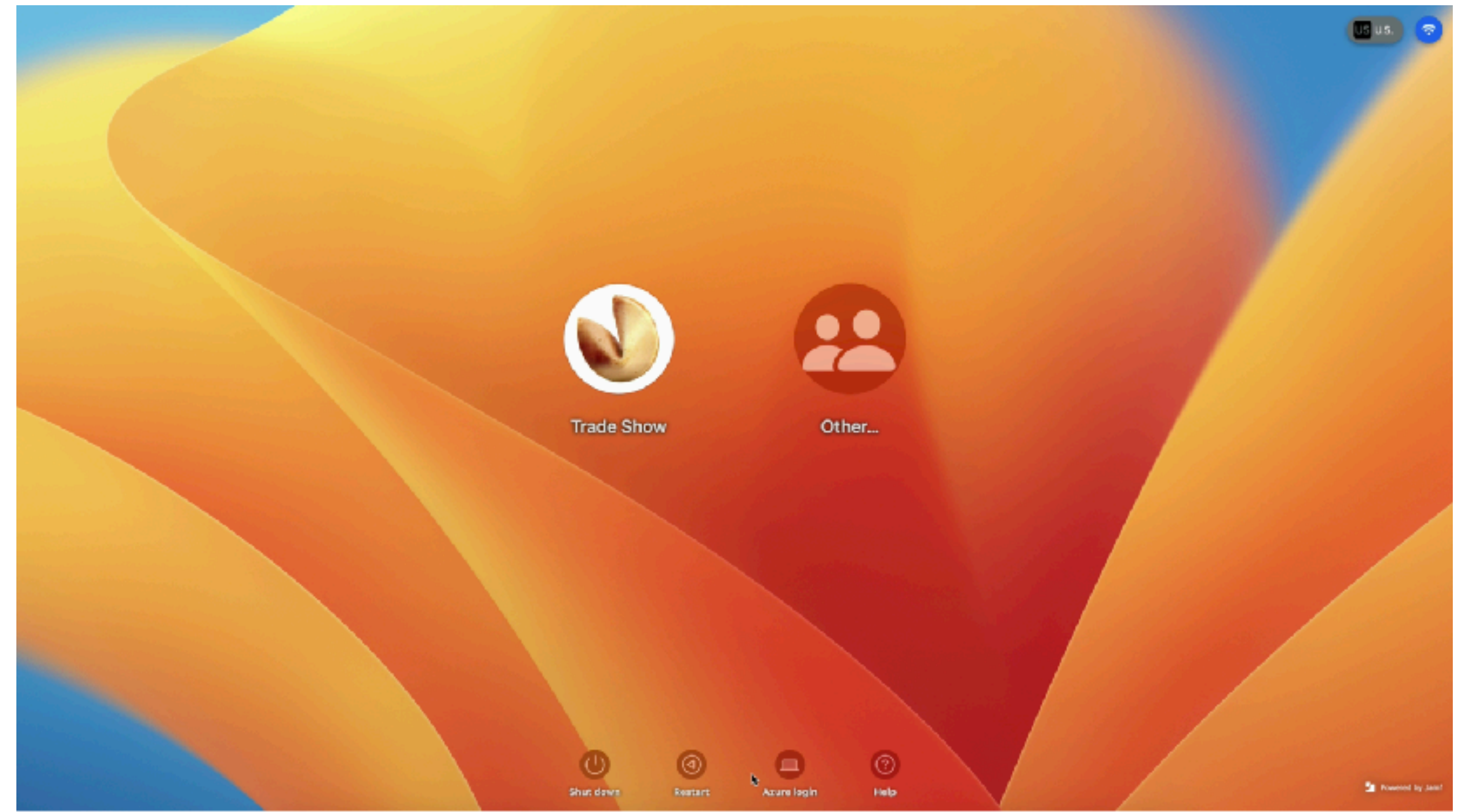
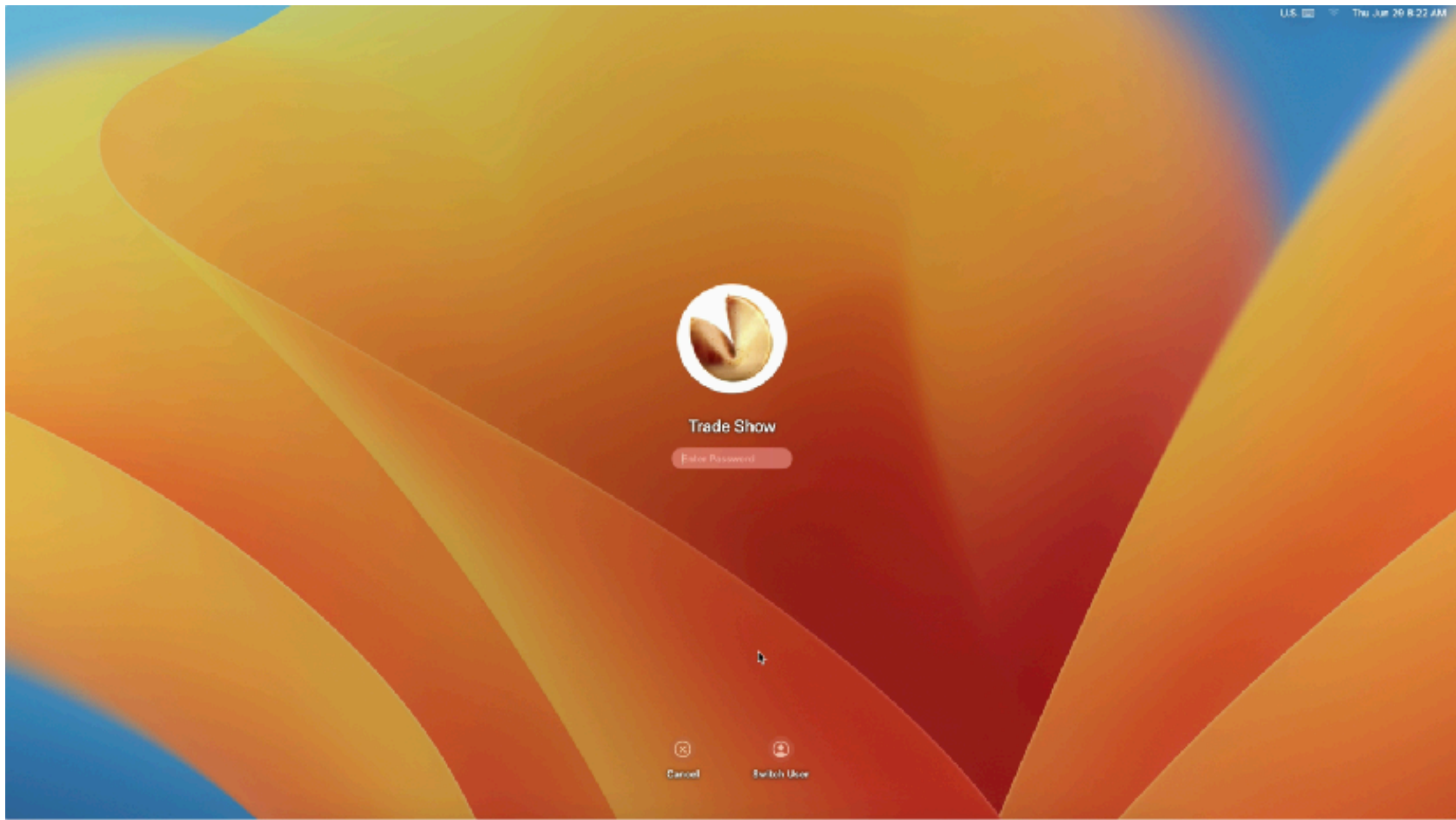
Restrict Allow



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



Other...



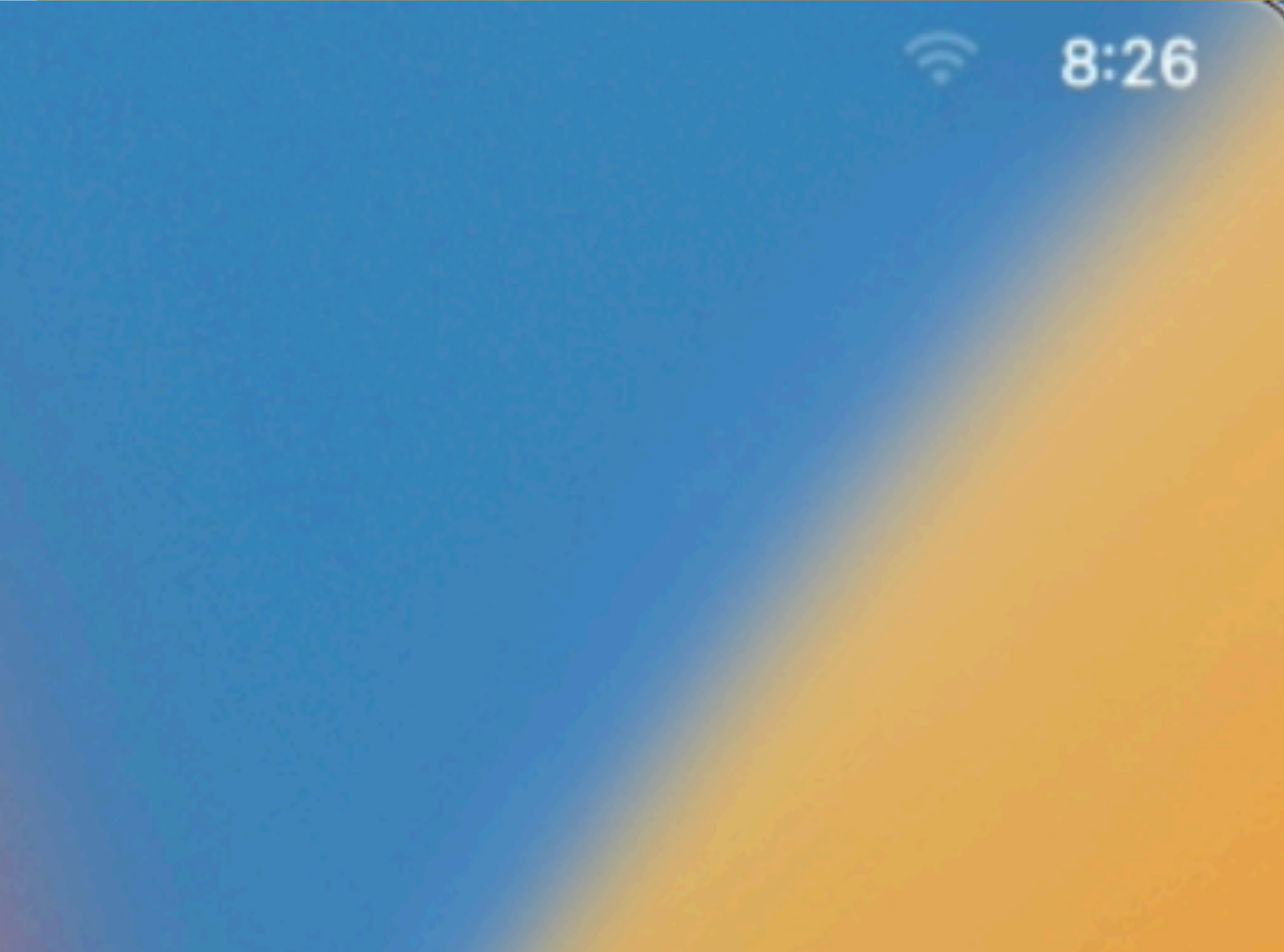
Shut Down



Restart



Sleep





Trade Show

Enter Password

⌵
Cancel

👤
Switch User

U.S. 🇺🇸 📶



Thu Jun 29 8:22 AM

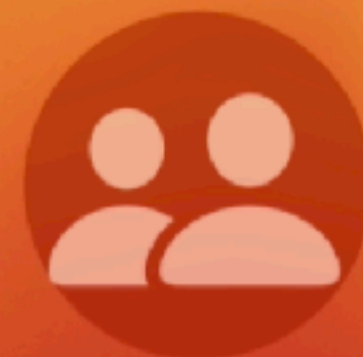


▶️

ABC 123



Trade Show



Other...



Shut down



Restart

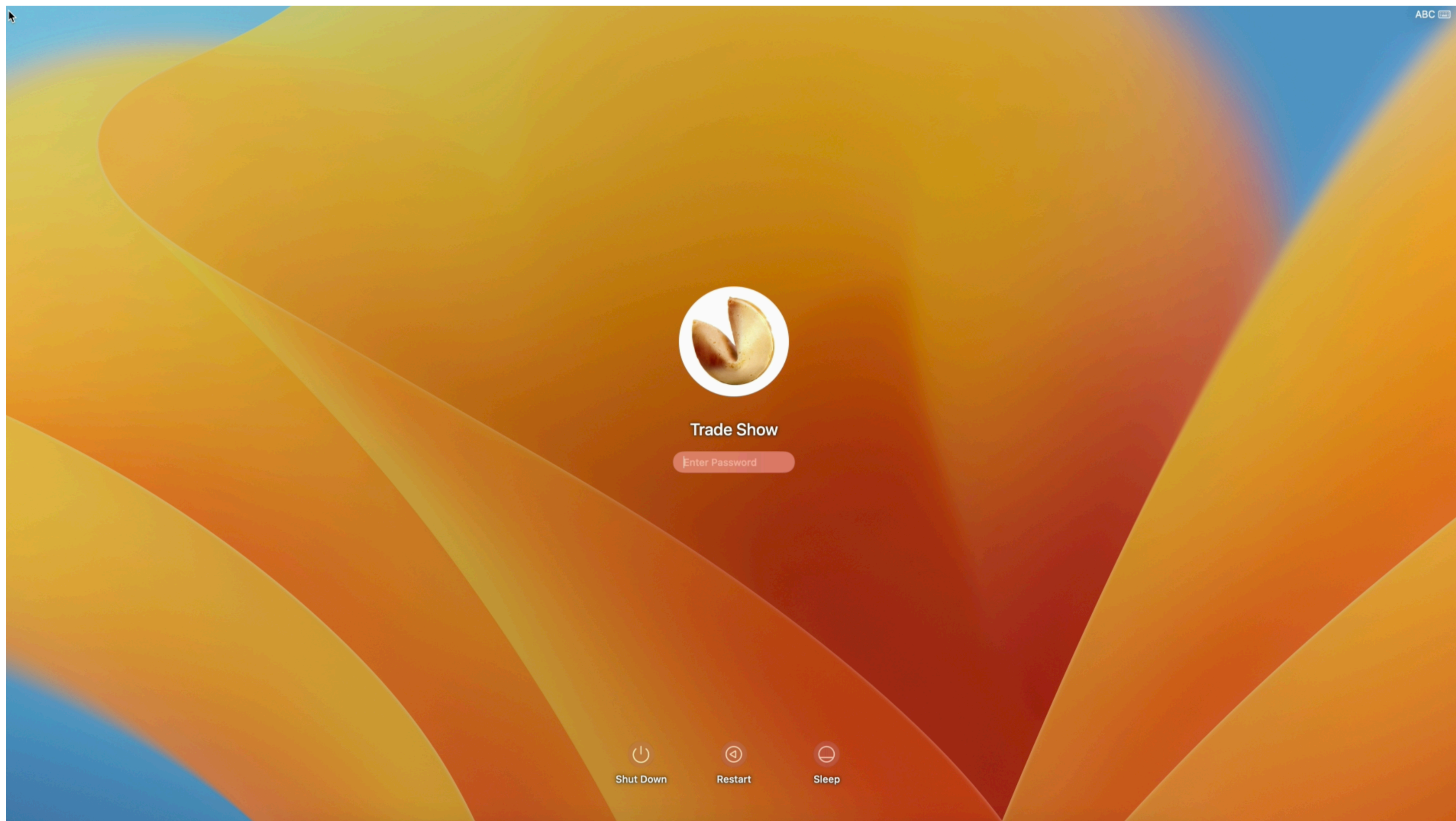


Azure login

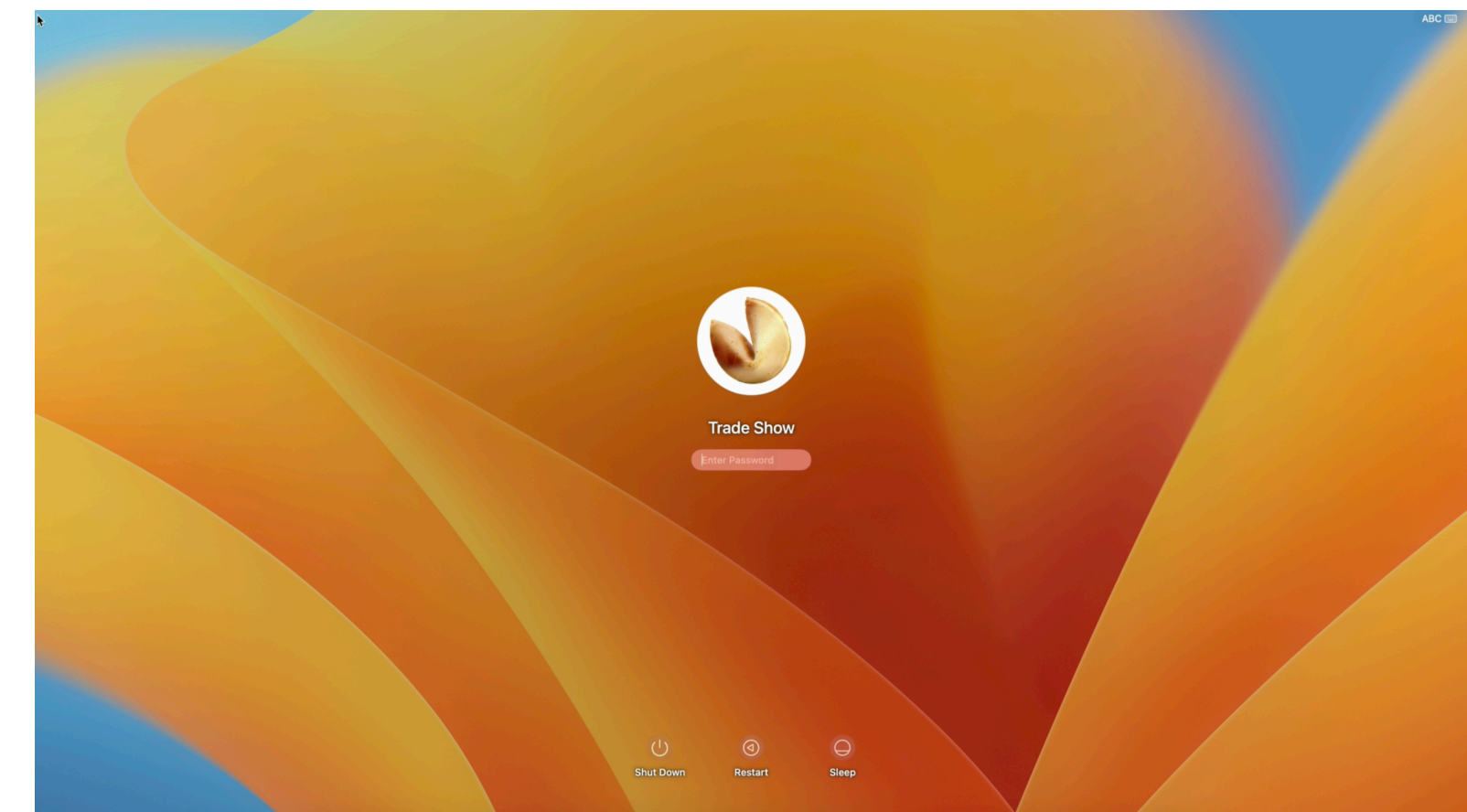
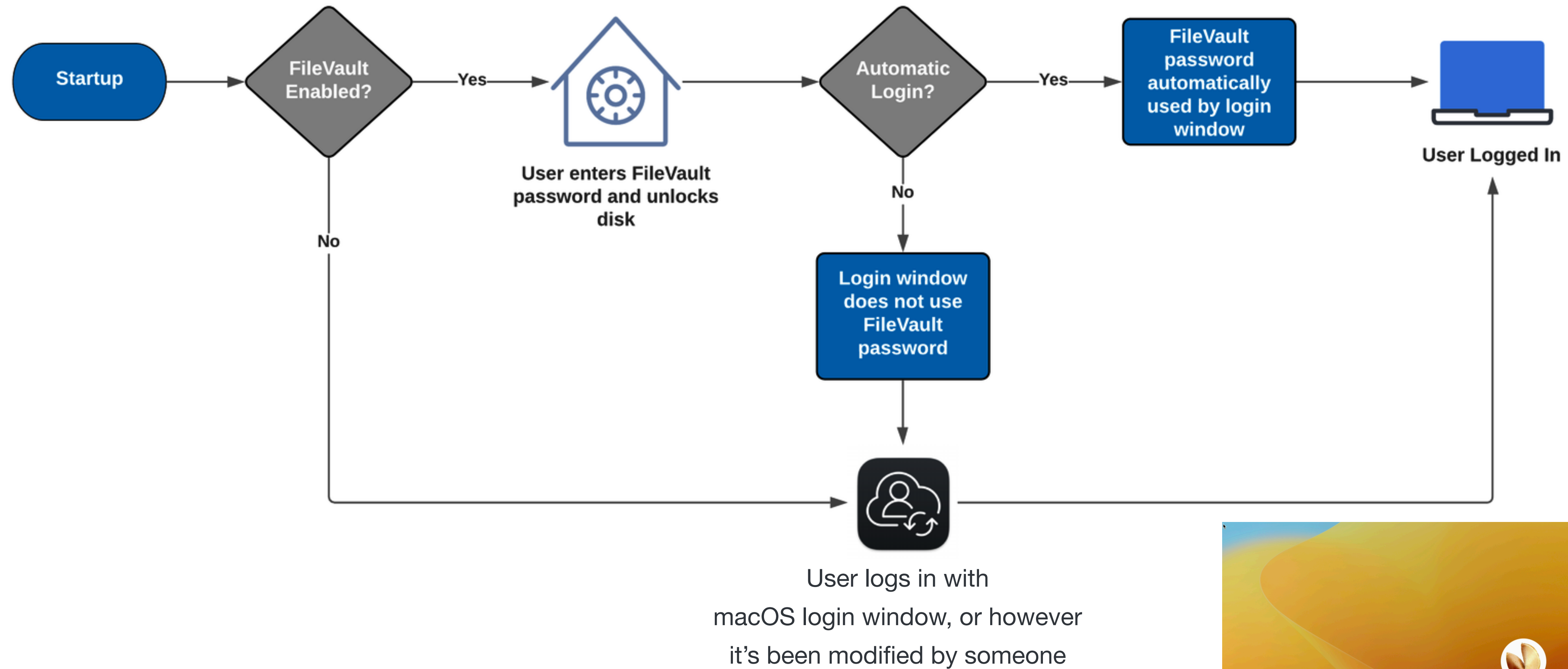


Help

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://hconline.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

Login Window - Alternatives



“Your password is...”

“My password is...”

Login Window - Alternatives



“Your password is...”

“I’m gonna make my password be...”

Login Window - Alternatives



“Your password is...”

“My password was...”

Login Window - Alternatives



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

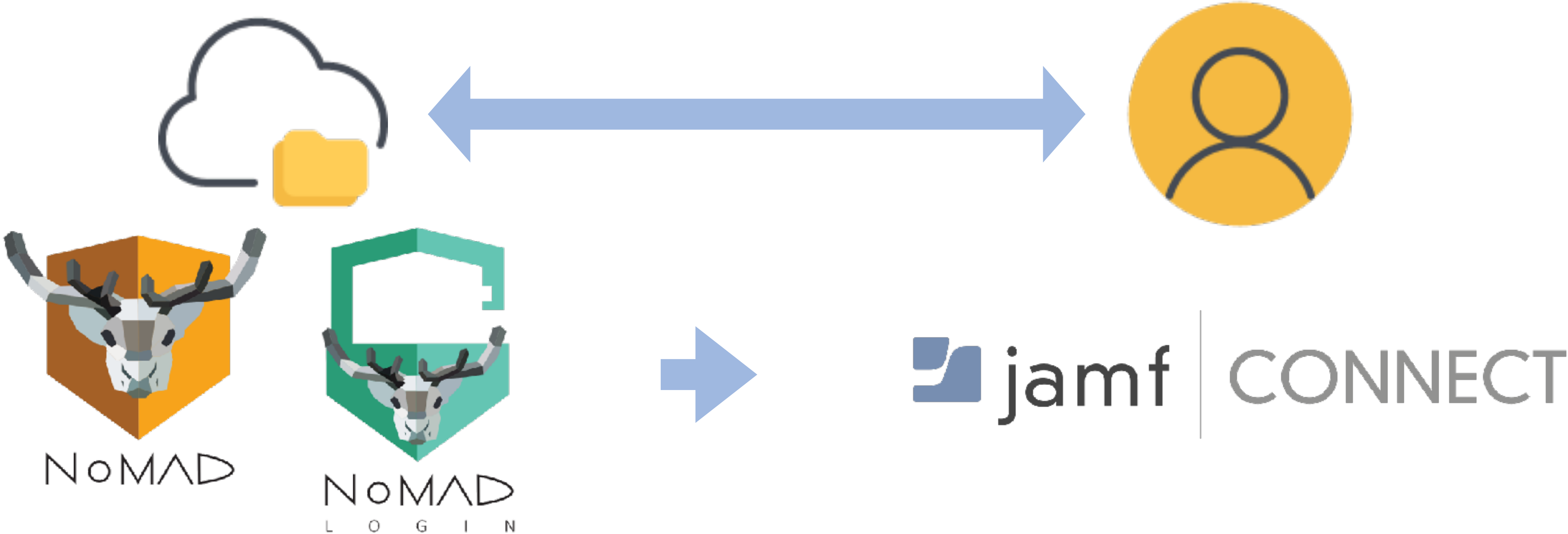
On-Premises Directory - Alternatives



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



Cloud Directory



Cloud Directory

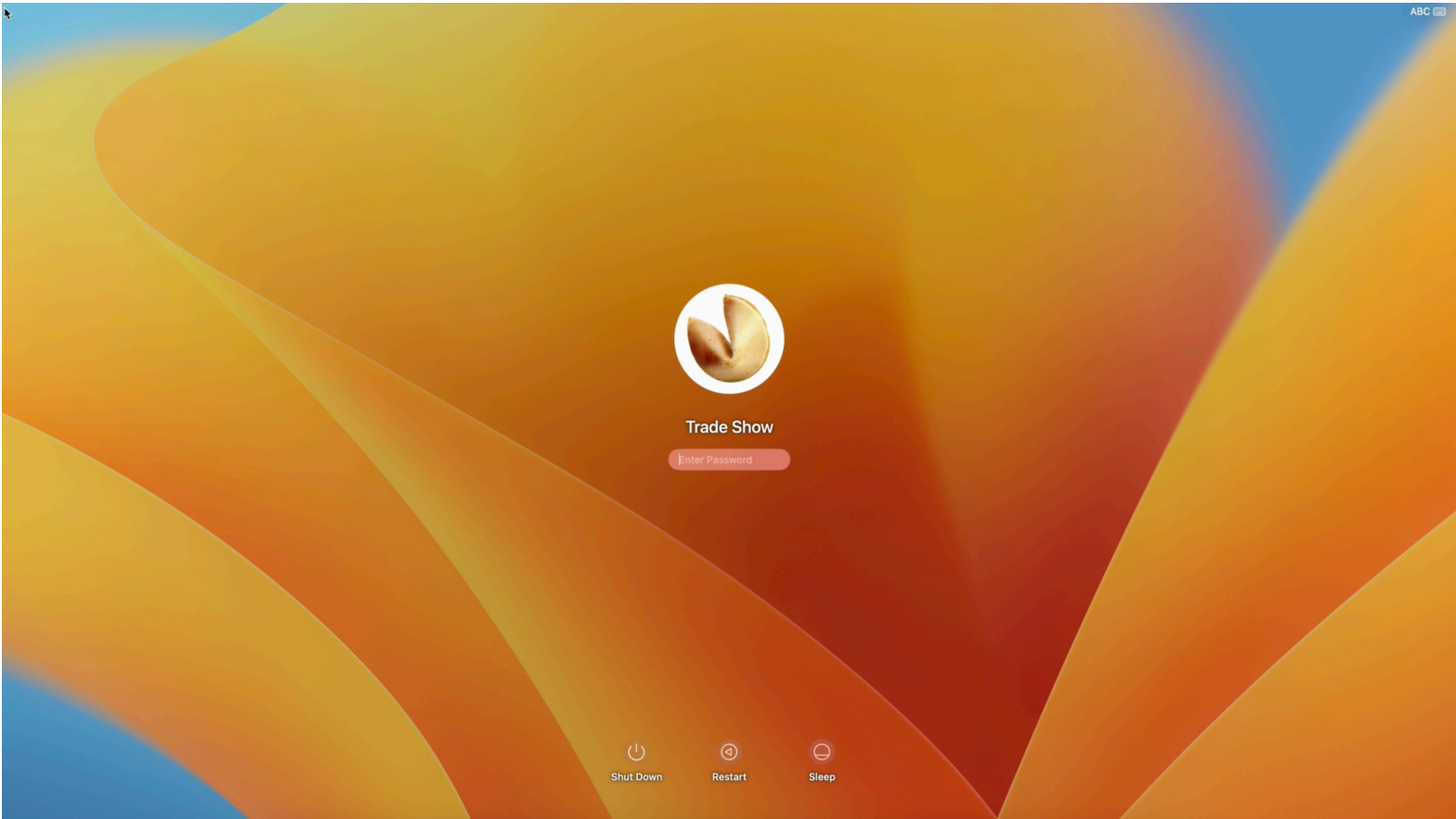
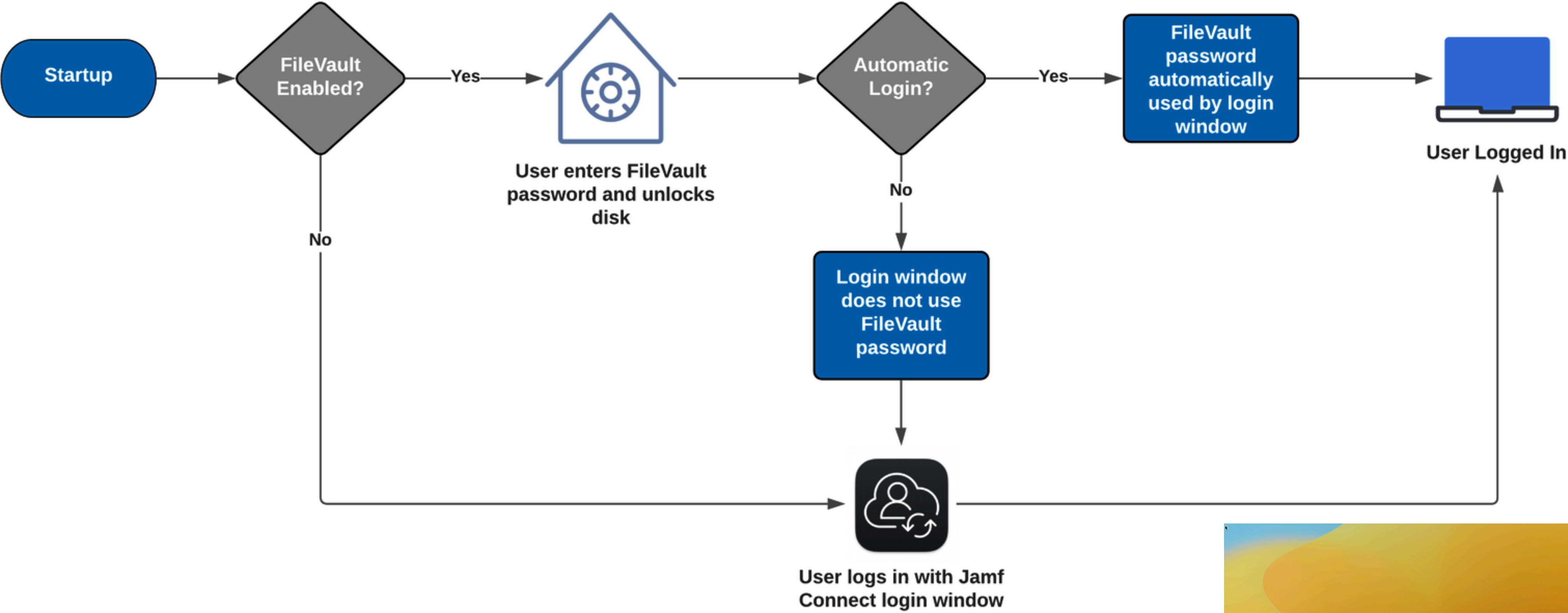


- Jamf Connect
- XCreds
- Mosyle Auth
- Kandji Passport

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

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/iamfHelper.app/Contents/MacOS/iamfHelper\

```

```

# 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	
<input type="button" value="▼"/>	Jamf Connect: Does a user deadpool file exist	is <input type="button" value="▼"/>	TRUE <input type="button" value="▼"/>	<input type="button" value="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 )
```

```
# 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     #####
```

```
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, updated speculation because some of this is in preview, it depends a LOT on identity providers, and macOS just had a change.

Single Sign-On Extension for Enterprise

← Microsoft Enterprise Single Sign-On Plug-in

Options Scope Show in Jamf Pro Dashboard

Search...

- General
- Application & Custom Settings 1 payload configured
- Single Sign-On Extensions 1 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

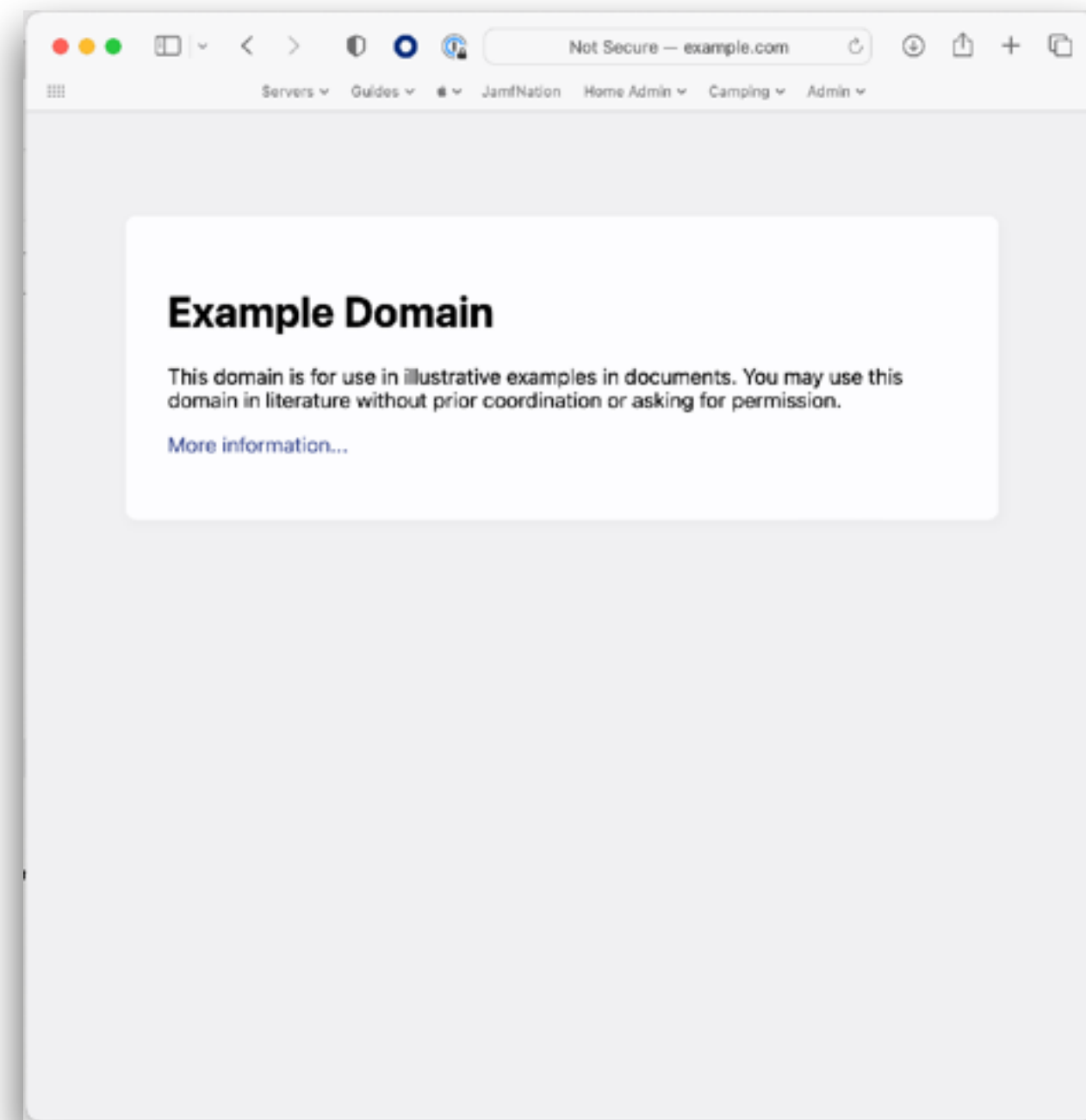
https://sts.windows.net

https://login.partner.microsoftonline.cn

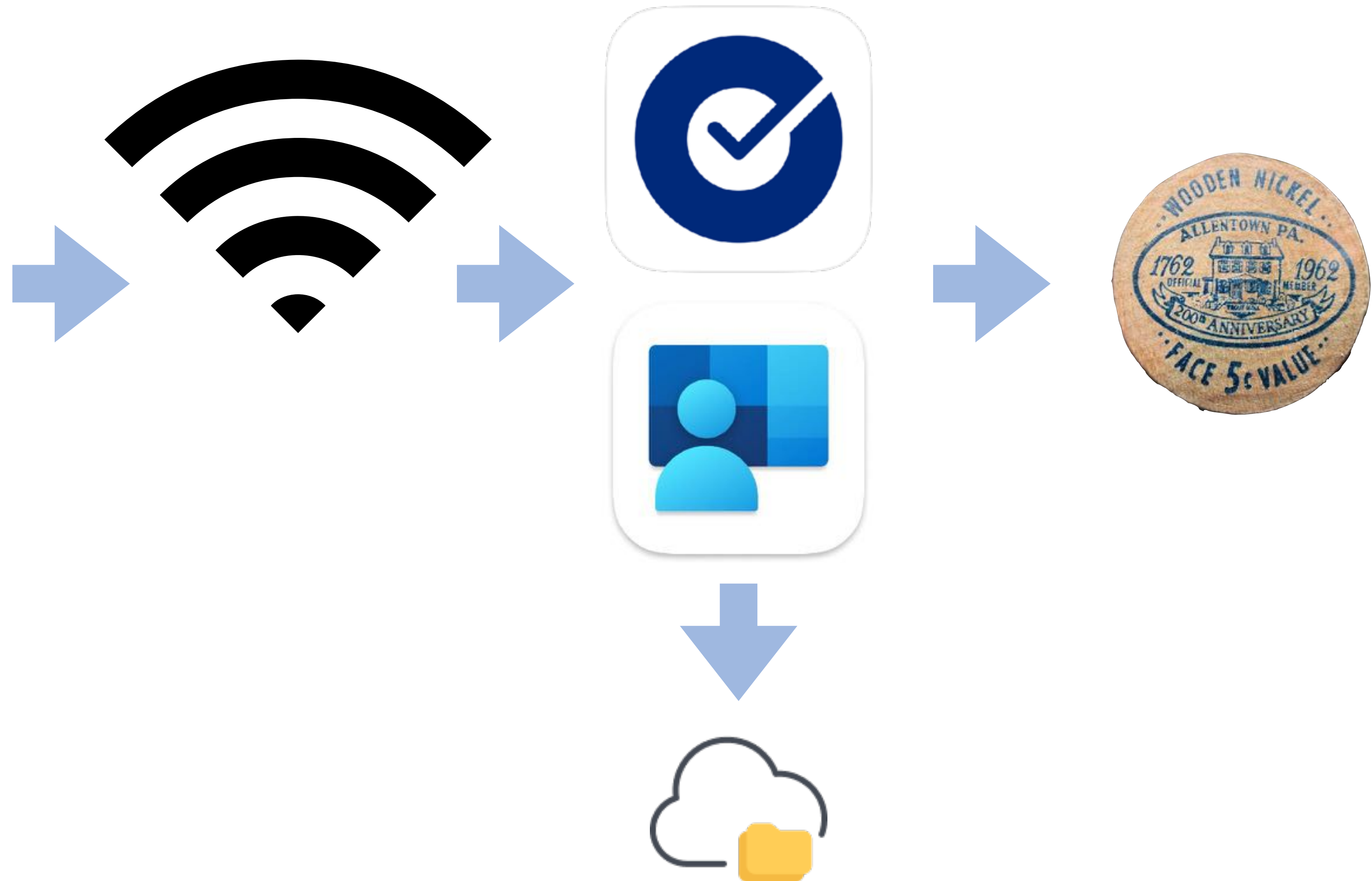
https://login.chinacloudapi.cn



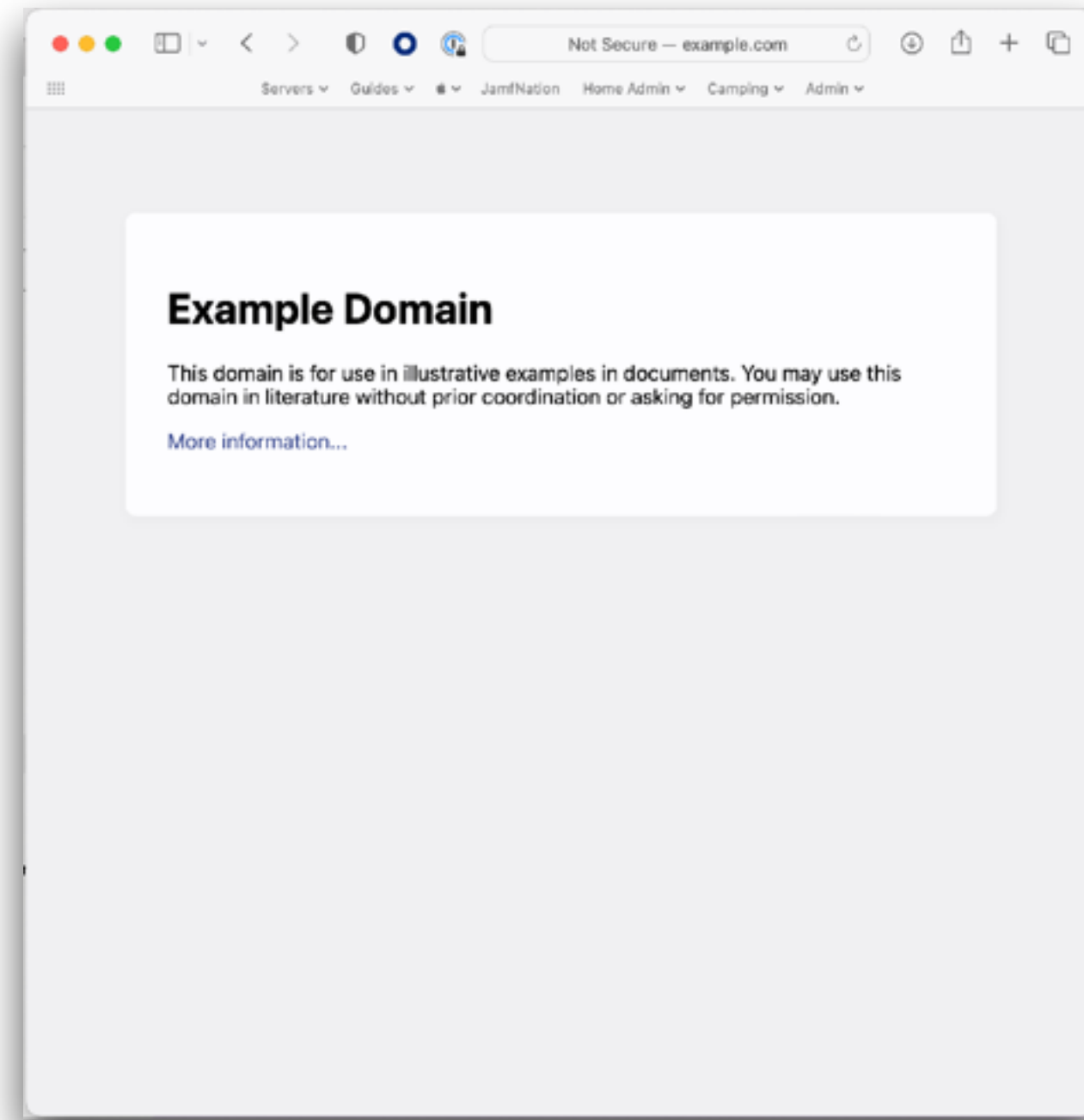
Single Sign-On Extension for Enterprise



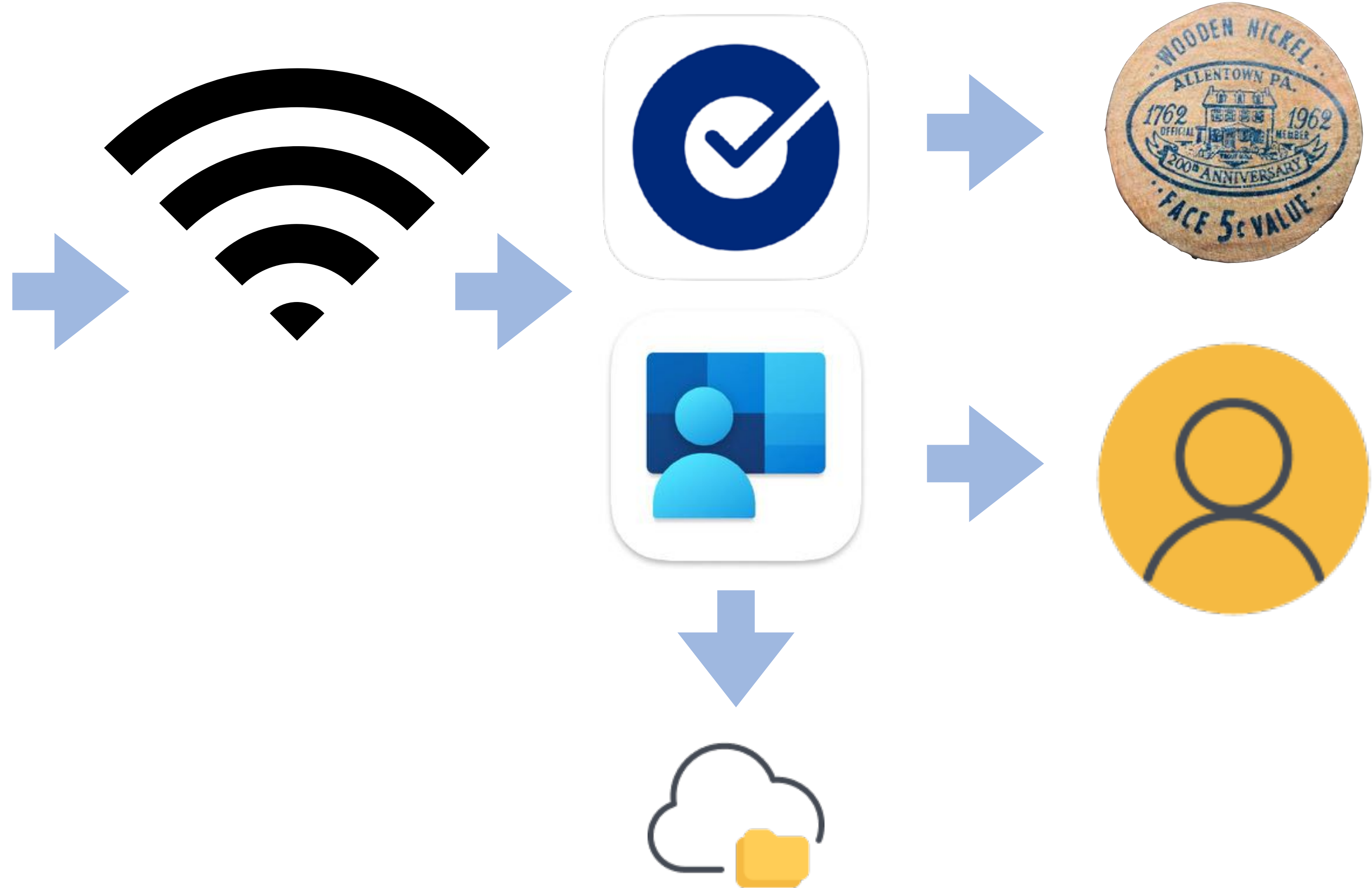
open <https://example.com/login>



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	⊘	

Content Warning:

No identity provider currently supports any of the stuff we're about to see.

Unless Michael is about to announce something cool in the next session.

Platform Single Sign On - as of macOS Sonoma

Feature	Platform Single Sign On	Jamf Connect & Others
Works at login window	✓	✓
Makes local user account	With HUGE caveats, ✓	✓
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 may still be required

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

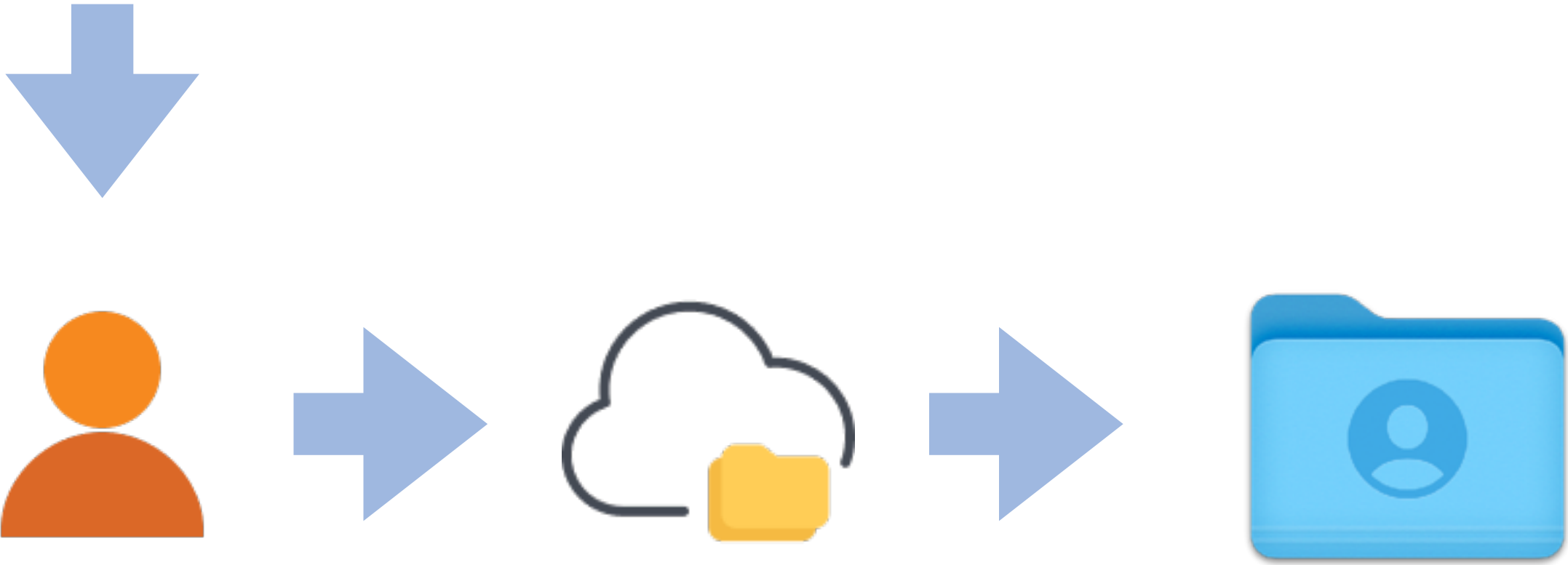


Platform Single Sign On - as of macOS Sonoma

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

Thank you.