jamf

NATION

User Conference

2018

Nathaniel Strauss

IT Manager
Shakopee Public Schools

Stop Binding Your Macs and Use NoMAD Login

Agenda:

- How we got here
- What is NoMAD Login?
- Security authorization database and authchanger
- Deploying NoMAD Login with Jamf and DEP
- Integrating with NoMAD

Local Accounts Good, AD Bad

	Local Account	Mobile Account	Network Account
Authority	Local Directory Service	LDAP	LDAP
Account Info	Local	Cached	Network
LDAP Binding	Not Required	Required	Required

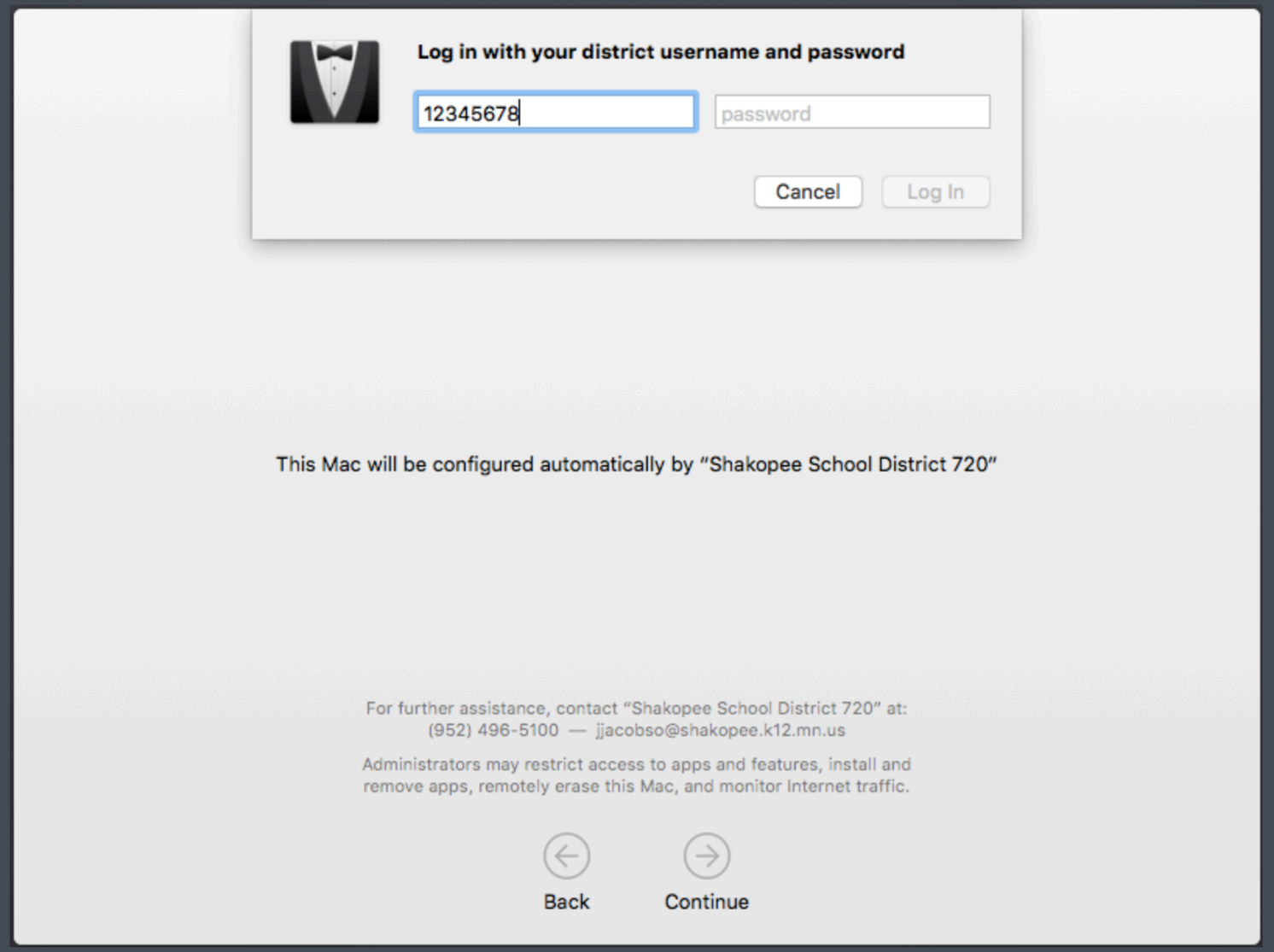
Why Local?

- About what authority controls the keys
- Avoid AD plugin
- No more keychain issues
- AD functionality without binding

DEP Setup Assistant Challenges

Allows authentication to MDM, but…

- Can’t force username
- Can’t force password
- Can’t guarantee either will match existing LDAP or identity management provider



Create a Computer Account

Fill out the following information to create your computer account.

Full name:

Account name:

This will be the name of your home folder.

Password: new password verify

Hint: optional

Set time zone based on current location

Back Continue

Moving to Local with DEP

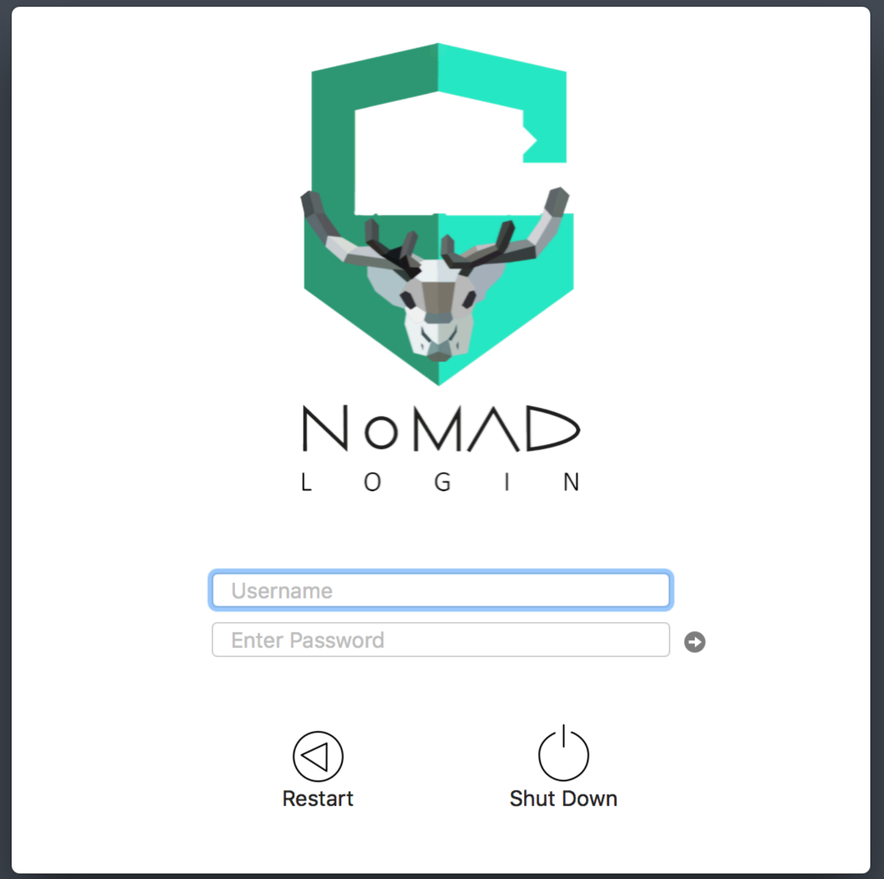
First attempt at creating local account using AD attributes…

- Captured username with DEP authentication
- Create LDAP EA to custom AD attribute with birthdate
- Grabbed birthdate and username with API
- Used sysadminctl to create a local account where the default password was based on username and birthdate.

Local Account Creation First Attempt

```
# Find UUID and use API call to get XML for computer record from Jamf Pro
uuid=$(system_profiler SPHardwareDataType | awk '/Hardware UUID/{print $3}')
data=$(curl -su $api_user:$api_pass $api_url/$uuid)

# xpath to get birthday LDAP extension attribute
birthdate=$(cat $data | xpath "//computer/extension_attributes/
extension_attribute" | grep -A2 "<name>birthdate</name>" | awk -F'<value>|</
value>' '{print $2}’)

# Create local account with default password based on birthdate
sysadminctl -addUser $user -fullName $full_name -password $default_password
```

Enter NoMAD Login



What is NoMAD Login?

Login window authorization plugin

- Creates local accounts based on AD users
- Replaces default login window
- Lives in /Library/Security/SecurityAgentPlugins
- Controlled by authorization mechanisms

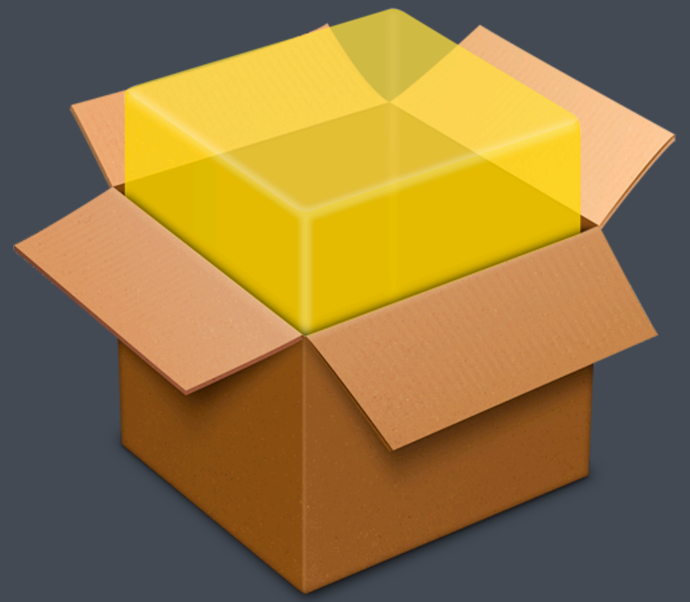
Deploying NoMAD Login

Package + Auth Mechs + Preferences =

Choosing a Package

When downloading there are two package options:

- NoMADLogin.pkg
- NoMADLogin-authchanger.pkg



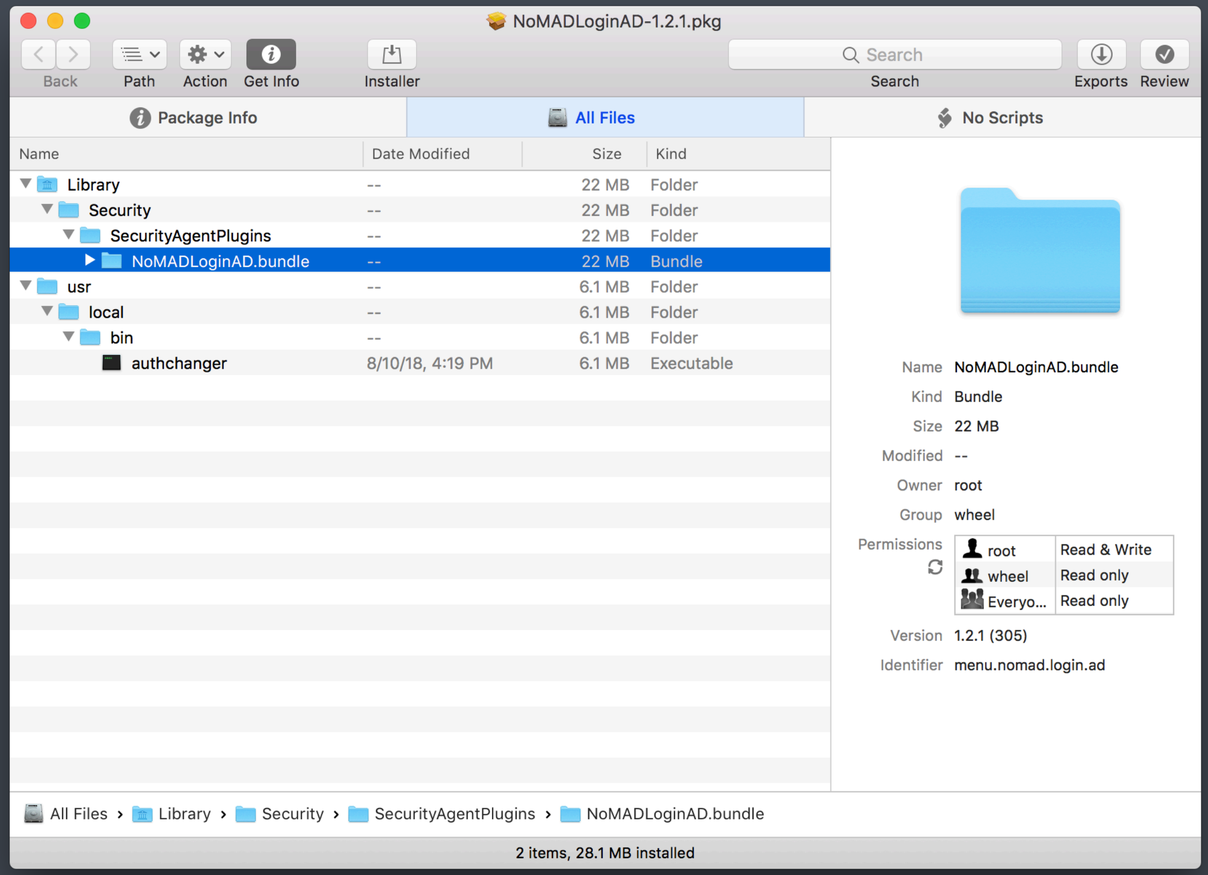
NoMADLogin-authchanger.pkg contains a postinstall script.

Choosing a Package Cont.

Both packages contain:

- NoMADLoginAD.bundle
- authchanger

“-authchanger.pkg” flavor postinstall script runs authchanger which modifies authorization mechanisms.



Authorization Mechanisms

An authorization mechanism is code that performs a step in the authorization process.

- Run in order at the login window
- One mechanism defines one function
- References code in NoMADLoginAD.bundle

```
$ security authorizationdb read system.login.console
$ authchanger -print
```

authchanger

- Utility to read and write authorization mechanisms
- Included in all pkgs
- Preferred over security commands

Auth Mechanisms - Default

```
$ authchanger -print  builtin:policy-banner
                      loginwindow:login
                      builtin:login-begin
                      builtin:reset-password,privileged
                      builtin:forward-login,privileged
                      builtin:auto-login,privileged
                      builtin:authenticate,privileged
                      PKINITMechanism:auth,privileged
                      builtin:login-success
                      loginwindow:success
                      loginwindow:FDESupport,privileged
                      HomeDirMechanism:login,privileged
                      HomeDirMechanism:status
                      MCXMechanism:login
                      CryptoTokenKit:login
                      loginwindow:done
```

Auth Mechanisms - NoMAD Login AD

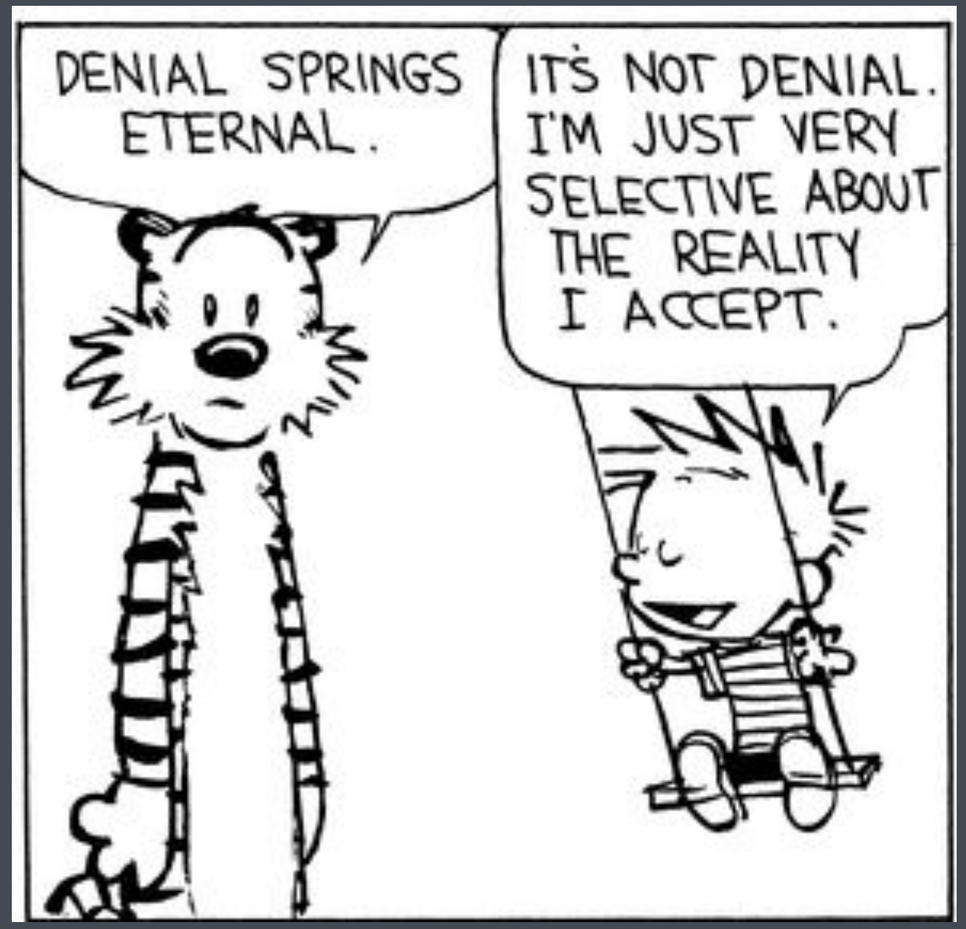
```
$ authchanger -AD
$ authchanger -print
```

```
builtin:policy-banner
NoMADLoginAD:CheckAD
NoMADLoginAD:EULA
NoMADLoginAD:PowerControl,privileged
NoMADLoginAD:CreateUser,privileged
NoMADLoginAD:DeMobilize,privileged
builtin:login-begin
builtin:reset-password,privileged
builtin:forward-login,privileged
builtin:auto-login,privileged
builtin:authenticate,privileged
PKINITMechanism:auth,privileged
builtin:login-success
loginwindow:success
loginwindow:FDESupport,privileged
HomeDirMechanism:login,privileged
HomeDirMechanism:status
MCXMechanism:login
CryptoTokenKit:login
loginwindow:done
NoMADLoginAD:EnableFDE,privileged
NoMADLoginAD:SierraFixes,privileged
NoMADLoginAD:KeychainAdd,privileged
```

Setting Preferences with Defaults

```
#!/bin/bash

# Variables
domain="mydomain.COM"
background_image="/Library/Desktop Pictures/High Sierra.jpg”
logo="/Library/Application Support/SPS/spslogo.png"

# Write default AD domain
defaults write /Library/Preferences/menu.nomad.login.ad.plist ADDomain $domain

# Set background image
defaults write /Library/Preferences/menu.nomad.login.ad.plist BackgroundImage
$background_image

# Set login window logo
defaults write /Library/Preferences/menu.nomad.login.ad.plist LoginLogo $logo
```

#imagingisdead



Jamf + NoLo Deployment Workflow

DEP Enrollment

Package

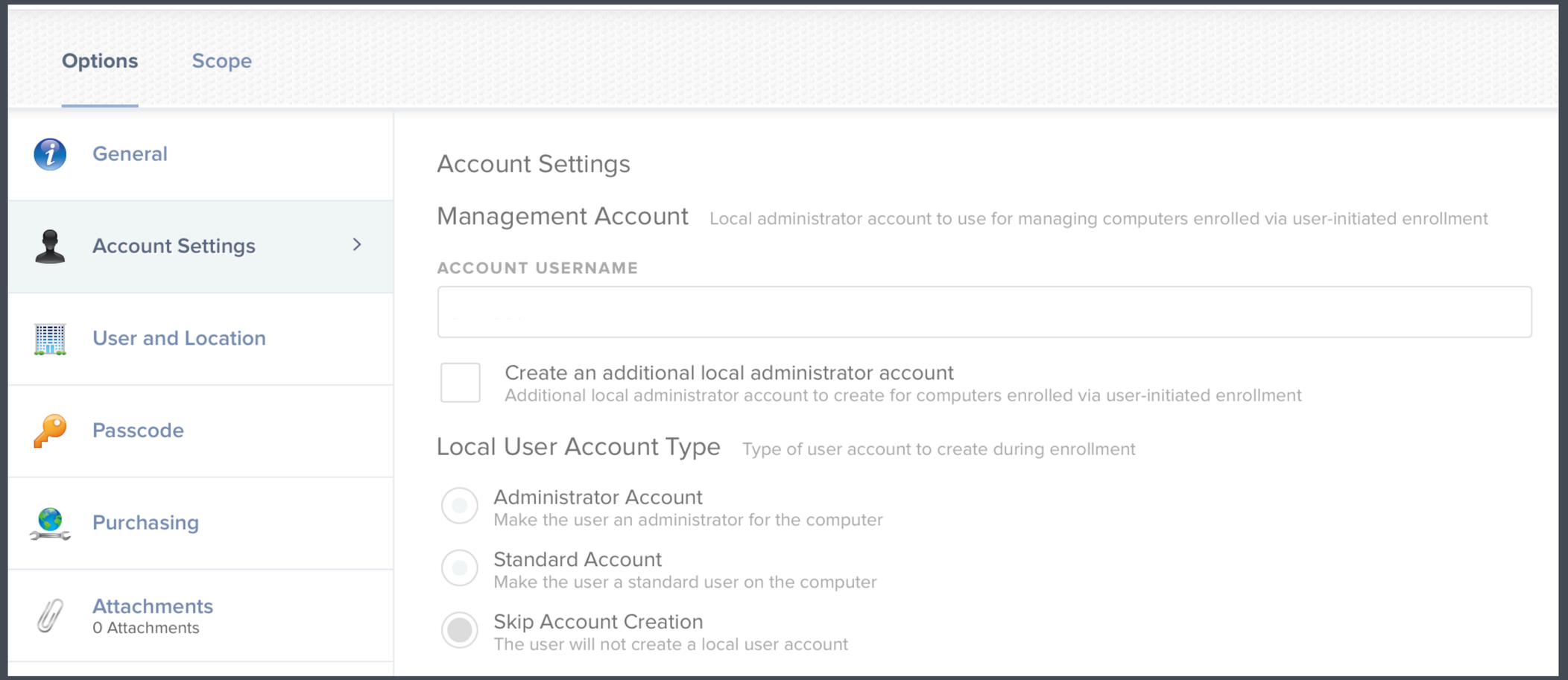
Preferences

Auth Mechanisms

Kill Login Window

NoMAD Login

PreStage Enrollment Settings



Postinstall Script

```
# Variables
domain="mydomain.COM"
background_image="/Library/Desktop Pictures/High Sierra.jpg”
logo="/Library/Application Support/SPS/spslogo.png"

# Write default AD domain
defaults write /Library/Preferences/menu.nomad.login.ad.plist ADDomain $domain

# Set background image
defaults write /Library/Preferences/menu.nomad.login.ad.plist BackgroundImage
$background_image

# Set login window logo
defaults write /Library/Preferences/menu.nomad.login.ad.plist LoginLogo $logo

# Set security authorization database mechanisms with authchanger
/usr/local/bin/authchanger -reset -AD

# Kill loginwindow process to force NoMAD Login to launch
/usr/bin/killall -HUP loginwindow
```

Enrollment Complete Policy

NAME	FREQUENCY	TRIGGER	SCOPE
DEP			
Launch NoMAD Login	Ongoing	Enrollment	No scope defined
1  Install NoMADLoginAD-1.2.1.pkg			
2  Run Script postinstallNoMADLogin.sh			

Considerations and Caveats

- Do I want to keep NoLo around after user provisioning?
  - authchanger -reset
- Mac must have a route back to AD for authentication to work.

Integrating with NoMAD

Securely pass a user's credentials to a keychain entry so that user is automatically signed into NoMAD.

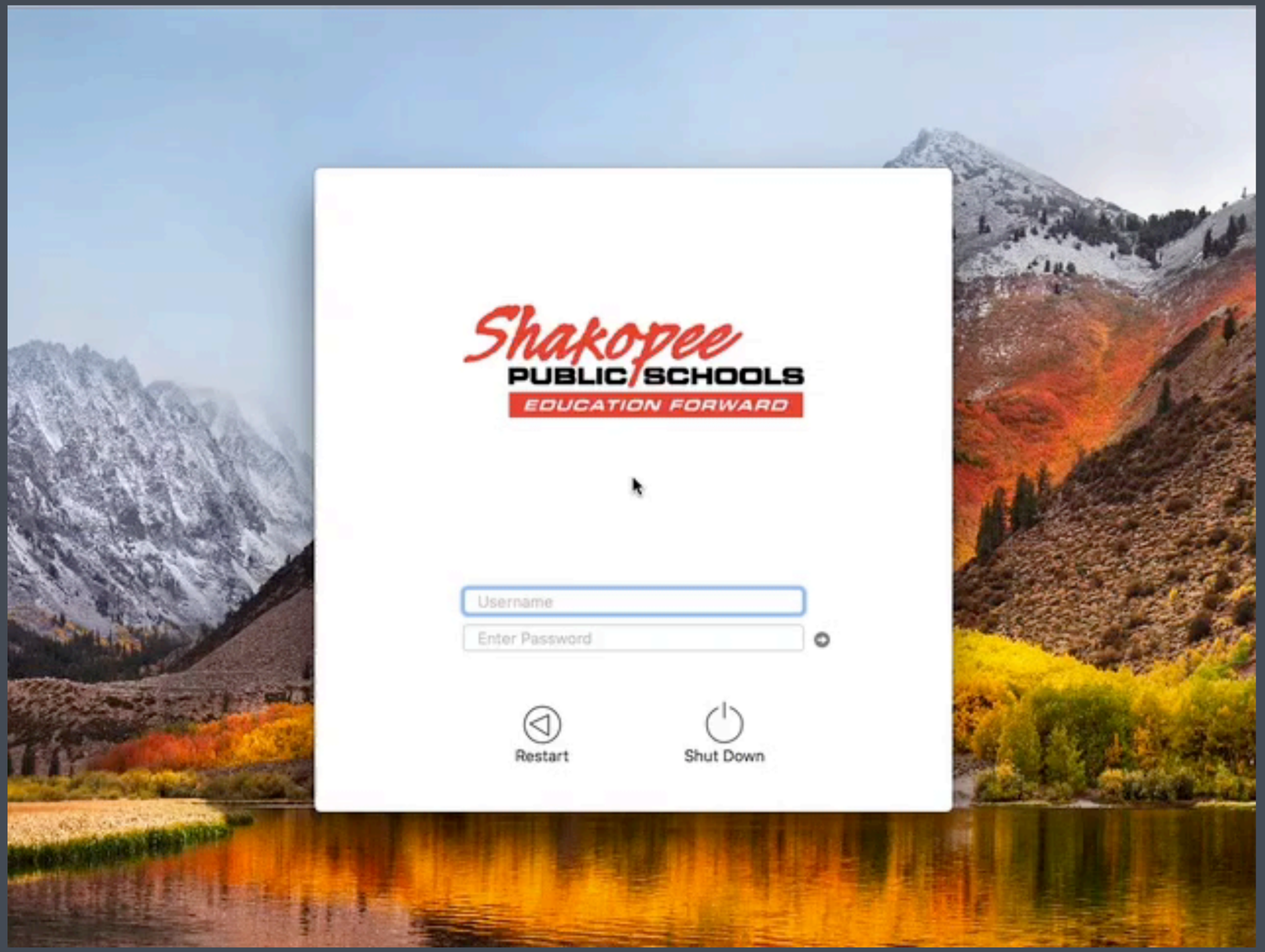
NoMAD **must** be installed before authentication.

Requires two additional preferences…

- KeychainAddNoMAD - adds keychain entry
- KeychainCreate - creates keychain if needed

Integrating with NoMAD Cont.

```
# Adds a NoMAD entry into the login keychain
defaults write /Library/Preferences/menu.nomad.login.ad.plist KeychainAddNoMAD -bool TRUE

# Should NoLo create a keychain if none exists
defaults write /Library/Preferences/menu.nomad.login.ad.plist KeychainCreate -bool TRUE
```



Jamf Connect

Paid versions of NoMAD and NoMAD Login with Okta support are now…

Open source is still open source.

Resources

- http://bit.ly/NoLo-JNUC2018
- https://nstrauss.info
- https://github.com/nstrauss

Thank you for listening!

Give us feedback by completing the 2-question session survey in the JNUC 2018 app.

UP NEXT

Session title

Session time