Just-in-time user... deletion?

Or, how I eliminated universally deployed admin accounts from my fleet and learned to love Jamf Connect

One of the great features of Jamf Connect is the ability to make a user account on demand simply by logging into the Mac. Jamf Connect will read an attribute from our identity provider to determine if a user should be an Administrator or get standard rights.

For our security conscious Mac Admins out there in the world (which should be all of you, I hope), this means that we can completely eliminate the "one ring to rule them all" type of admin accounts deployed to the fleet, usually stuck with some "secret" password that everyone in the company ends up knowing eventually. (I'm looking at you, `Jamf1234`.)

Now this is great, but then we run into trouble - we have a user account on a machine that we just needed for 5 minutes to fix a one-off type of problem, and in two years when we go back to that machine to fix another random one-off problem, now we have a user account where the admin has NO idea what the local user password is and everything explodes.

Until now.

What the workflow does:

1. An administrator makes an account just-in-time with the Jamf Connect login mechanism. Could be a one-off fix, could be resetting a forgotten local password. Whatever it is, admin is done, time to clean up like a good Scout.
2. The administrator opens Jamf Self Service and runs a Policy - this runs a script that looks for any account created by Jamf Connect in the last 60 minutes (which you can adjust), and drops a touchfile into something like `/Library/Application Support/JAMF/Receipts` (though you could change that to `/private/tmp` if you wanted) with a list of local short names that need to be deleted. The script then runs a `jamf recon` command to update the computer inventory record with...

3. An extension attribute that looks for the existence of this list of users in the deadpool. This extension attribute is the target of…
4. A Smart Computer Group which has all the computers with this deadpool file that exists which is the target of a scope of…
5. A Policy which is set to run with an Execution Frequency of "Ongoing", a trigger of "Reoccuring Check-in", and scoped to the Smart Computer Group above which will run a script that…
6. Looks for the deadpool list, runs a `jamf deleteAccount` command for every user in the list, moves the deadpool list out to a separate file to make sure the script ran, and runs another `jamf recon` command to clear the extension attribute that removes the computer from the scope of the policy.

Known Gotchas:

Only One Admin Account:

When Apple introduced macOS Big Sur, they changed how FileVault securetokens can be distributed to users. Big Sur was the first OS that allowed there to be a standard user account created as the first user AND that user would receive a securetoken to decrypt the machine without the nightmare fuel of binding the Mac to a directory service like Open Directory or Active Directory.

Being a security conscious IT professional, you probably want to follow the CIS guidelines and limit access to administrator rights only when they're needed. So chances are pretty good you used Jamf Connect or your Automated Device Enrollment prestage and created the first user account as a standard account. Safest admin account on a box is one that doesn't exist until you need it, right?

As of macOS Monterey 12.1, macOS blocks you from deleting an administrator account with a securetoken if it's the only admin account with a securetoken on the box. WAIT WAT?

Yep, even though the MDM can hand out securetokens to new users like candy thanks to the wonders of the bootstrap token, macOS is going back to its Catalina and earlier way of "protecting you from yourself" and preventing the admin account from deletion.

The Workaround:

It looks like just about ANY tool that elevates a standard user to an admin user, even just for a short period of time, will solve this problem.

You could…
A) Good thing you use Jamf Connect! In your identity provider, temporarily move the standard account user to a role that grants them administrator rights on the Mac. Tada, now there's two admins as soon as that user logs out and logs back in. Now, run the policy to nuke the other admin account and do a re-occurring checkin. The temporary admin account is deleted as expected! Last, move the user back to a standard account in the identity provider, log out, log back in, and they're a standard user again.

B) Install the macOS Enterprise Privileges app from https://github.com/SAP/macOS-enterprise-privileges. With one self service policy, a user could be elevated to an admin with the Privileges app command line interface, run the cleanup script, trigger a re-occurring check-in, and be bounced back to a standard account. For extra security, toss in DEPNotify or SplashBuddy in full screen mode to prevent the user from performing "unexpected activities."

C) Use the https://github.com/jamf/MakeMeAnAdmin script (maybe with a shortening of the time down to 5 minutes or so) in combination with the Self Service script to nuke users and run an re-occurring check-in.

The Scripts

Look for users created in the last 60 minutes

Upload this script to Jamf Pro. Create a Policy with Execution Frequency set to "Ongoing", no Trigger, scoped to all computers (and consider gating this behind a Self Service login requirement, maybe just for the IT team). Add the Script payload to run the uploaded script. Go to the Self Service tab and allow the policy to be run from Self Service with an appropriate description and warnings.

```
#!/bin/bash


# PART ONE: Find a list of Jamf Connect users who were created in the last X
#                         minutes
```

```
#
# WHY: Jamf Connect allows for just-in-time account creation on a macOS client.
# So this means that an admin may want to just pop in and do some magic, log out
# and go away.  But there's no easy way to clean up after yourself, and like any
# good Girl Scout, we should always leave our campsite better than when we found
# it.
#
# HOW: Upload this script into Jamf Pro.  Create a policy to run the script with
# and ongoing excution frequency and set to run via Self Service.  It may make
# sense to restrict the app to specific users and require that IT folks sign in
# to self service to prevent some users from running the script.
#
# WHAT: We'll search all the accounts that have passwords, see if it was created
# with Jamf Connect, determine if the account was created within the last X
# minutes (you can adjust the number below).  If yes, will be added to a space
# delimited list of user account short names to be written to
# /private/tmp/.userCleanup (which you can also adjust below).
#
# Combine this with an extension attribute to read that file, a Smart Computer
# Group to drop machines into a target group to run a policy at reoccuring
# check-in, and a policy that reads that file and runs a jamf deleteAccount
# command to kill that account.
#
# — SRABBITT 21DEC2021

# MIT License
#
# Copyright (c) 2021 Jamf Software

# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.

#look for users created in the last X minutes
userAge=60

# Touch file with list of users to be deleted
DELETE_USER_TOUCH_FILE="/Library/Application Support/JAMF/Receipts/.userCleanup"
# Credit: Steve Wood

# Location of the Jamf binary
JAMF_BINARY="/usr/local/bin/jamf"

# Declare list of unmigrated users variable
listOfUsers=""

# Warn users of what is going to happen
responseCode=$(/Library/Application\ Support/JAMF/bin/jamfHelper.app/Contents/MacOS/jamfHelper\
        -heading "WARNING - THIS APPLICATION CAN DELETE USER DATA" \
        -cancelButton 1\
        -button2 "Continue"\
        -button1 "ABORT"\
        -windowType utility\
        -description "This application will search for user accounts created in the last $userAge minutes.  It will mark those accounts for deletion which will happen within the next check-in period to Jamf Pro.  If you do NOT want to continue, press ABORT."\
        -title "Jamf Connect Cleanup Script"\
        -icon "/System/Library/CoreServices/CoreTypes.bundle/Contents/Resources/ToolbarDeleteIcon.icns")

# If a user hits the abort button, get ouf the script and declare an exit code
# of 999.  Policy will show as a failure in Jamf Pro logs.
if [[ $responseCode = 0 ]]; then
        exit 999;
fi

# Convert userAge to seconds
userAge=$((userAge * 60))

# For all users who have a password on this machine (eliminates service accounts
# but includes the _mbsetupuser and Jamf management accounts...)
for user in $(dscl . list /Users Password | awk '$2 != "*" {print $1}'); do
        # If a user has the attribute "OIDCProvider" in their user record, they are
        # a Jamf Connect user.
        MIGRATESTATUS=($(dscl . -read /Users/$user | grep "OIDCProvider: " | awk {'print $2'}))
        # If we didn't get a result, the variable is empty.  Thus that user is not
        # a Jamf Connect Login user.
        if [[ -z $MIGRATESTATUS ]];
                then
                        # user is not a jamf connect user
                        echo "$user is Not a Jamf Connect User"
                else
                        #Thank you, Allen Golbig.
                        create_time=$(dscl . -readpl /Users/$user accountPolicyData creationTime | awk '{ print $NF }')

                        # Strip the annoying float and make it an int
                        create_time=$( printf "%.0f" $create_time )


                        # Get the current time in Epoch format
                        start_time=$(date +%s)


                        # Remove the userAge number of seconds that we're looking for....
                        start_time=$(( start_time - userAge ))


                        # If the user account was created AFTER the current time minus X
                        # minutes, add the user UNIX short name to a list of users.
                        if (( $start_time < $create_time));
                        then
                                listOfUsers+=$(echo "$user ")
                        fi
                fi
done

# If we didn't find anything, either our admin took a lot longer than 60 minutes
# to fix the problem or something else went wrong.
if [[ -z $listOfUsers ]];
        then
                /Library/Application\ Support/JAMF/bin/jamfHelper.app/Contents/MacOS/jamfHelper\
                -heading "ERROR - No users found"\
                -button1 "Continue" \
                -windowType utility \
                -description "No local user accounts were created with Jamf Connect Login in the last $userAge seconds.  User account may need to be deleted manually." \
                -title "Jamf Connect Cleanup Script" \
                -icon "/System/Library/CoreServices/CoreTypes.bundle/Contents/Resources/ProblemReport.icns"
        else
                # Otherwise, we found someone - time to tell the user that it's
                # curtains... lacy, wafting curtains for that user.
###
### YOU CAN EDIT THIS WARNING MESSAGE TO LOCALIZE FOR YOUR IT TEAM HERE
###
                warningMessage="The following accounts will be deleted within 15 minutes of this policy running:

$listOfUsers
```

```
Press ABORT to stop."

                # Give users one last chance to avoid the end times for that user...
                responseCode=$(/Library/Application\ Support/JAMF/bin/jamfHelper.app/Contents/MacOS/jamfHelper \
                        -icon "/System/Library/CoreServices/CoreTypes.bundle/Contents/Resources/AlertStopIcon.icns" \
                        -button2 "Continue" \
                        -description "$warningMessage" \
                        -heading "User Account Deletion" \
                        -windowType utility \
                        -cancelButton 1 \
                        -button1 "ABORT" \
                        -title "WARNING - POTENTIAL FOR DATA LOSS")

        # If a user hits the abort button, get ouf the script and declare an exit
        # code of 666  Policy will show as a failure in Jamf Pro logs.
        if [[ $responseCode = 0 ]]; then
                exit 666;
        fi
                /Library/Application\ Support/JAMF/bin/jamfHelper.app/Contents/MacOS/jamfHelper \
                -heading "How to abort" \
                -button1 "Continue" \
                -windowType utility \
                -description "If you change your mind, delete the file located at $DELETE_USER_TOUCH_FILE immediately." \
                -title "Jamf Connect Cleanup Script"  \
                -icon "/System/Library/CoreServices/CoreTypes.bundle/Contents/Resources/AlertStopIcon.icns"
fi

# Write the list of doomed users to the doomed user file.
echo "$listOfUsers" > "$DELETE_USER_TOUCH_FILE"

# Run a recon so we update the extension attribute
# and alert Jamf Pro that this list exists
$JAMF_BINARY recon
```

Create an extension attribute to check for the deadpool list

Navigate to Jamf Pro settings → Computer Management → Extension Attributes. Create a new EA based on the result of a script and upload the following script.

```
#!/bin/bash

# PART TWO: Extension Attribute - Does the deadpool file exist
#
# WHY: Jamf Connect allows for just-in-time account creation on a macOS client.
# So this means that an admin may want to just pop in and do some magic, log out
# and go away.  But there's no easy way to clean up after yourself, and like any
# good Girl Scout, we should always leave our campsite better than when we found
# it.
# HOW:         1) Create an extension attribute with this script to look for the
#                        presence of file located at $DELETE_USER_TOUCH_FILE (defined below)
#                3) Create a Smart Computer Group based on the extension attribute above
#                        and that the file exists
#                4) Create a policy that runs at reoccuring checkin, scoped to computers
#                        that belong to the Smart Computer Group above.  Tell the policy to
#                        the delete a list of users script..

# WHAT: EA will return "TRUE" if the deadpool file user list exists.

# MIT License
#
# Copyright (c) 2021 Jamf Software

# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
```

```
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.

# — SRABBITT 21DEC2021


# Location of user deadpool list
DELETE_USER_TOUCH_FILE="/Library/Application\ Support/JAMF/Receipts/.userCleanup"

if [ -f "$DELETE_USER_TOUCH_FILE" ]; then
        echo "<result>TRUE</result>"
else
        echo "<result>FALSE</result>"
fi
```

Create a Smart Computer Group

Create a Smart Computer Group to list all computers where the extension attribute returned above is `TRUE`.

Create a policy to delete the deadpool users

Upload the following script to Jamf Pro. Create a new policy, Ongoing execution frequency, Trigger set to reoccurring check-in, scope set to only computers in the Smart Computer Group you defined above. Payload will be the script to run to clean up users.

```
#!/bin/bash

# PART THREE: Delete a list of users
#
# WHY: Jamf Connect allows for just-in-time account creation on a macOS client.
# So this means that an admin may want to just pop in and do some magic, log out
# and go away.  But there's no easy way to clean up after yourself, and like any
# good Girl Scout, we should always leave our campsite better than when we found
# it.

# HOW:         1) Upload this script into Jamf Pro.
#                 2) Create an extension attribute to look for the presence of a file
#                        located at $DELETE_USER_TOUCH_FILE (defined below)
#                3) Create a Smart Computer Group based on the extension attribute above
#                        and that the file exists
#                4) Create a policy that runs at reoccuring checkin, scoped to computers
#                        that belong to the Smart Computer Group above.  Tell the policy to
#                        run this script.

### NOTE: THE JAMF BINARY COMMAND TO DELETE USERS IS COMMENTED OUT BELOW.  YOU MUST UNCOMMENT THIS.  POTENTIAL FOR DATA LOSS!!! ###

# WHAT: Script will read the list of users at $DELETE_USER_TOUCH_FILE and run
#                the jamf binary command to delete that user and all its home directory
#                data.

# MIT License
#
# Copyright (c) 2021 Jamf Software

# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.

# — SRABBITT 21DEC2021

# Location of user deadpool list
DELETE_USER_TOUCH_FILE="/Library/Application Support/JAMF/Receipts/.userCleanup"
# Credit: Steve Wood

# Location of the user deadpool list after running script (confirmation file
# for auditing)
CONFIRM_USER_TOUCH_FILE="/private/tmp/.userDeleted"

# Location of the Jamf binary
JAMF_BINARY="/usr/local/bin/jamf"
```

```
# Convert the space separated list of users into an array for looping through
listOfUsers=$(cat "$DELETE_USER_TOUCH_FILE")
arrayOfUsers=($listOfUsers)

# For every user in the list, delete the user account with the Jamf binary
for user in ${arrayOfUsers[@]}; do

        ############################################################################
        ############################################################################
        ### 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
done

# Move the delete file for auditing purposes
/bin/mv "$DELETE_USER_TOUCH_FILE" "$CONFIRM_USER_TOUCH_FILE"
```

```
# Run a recon to clear out the extension attribute / smart computer group for
# running this process.
$JAMF_BINARY recon
```