

Management Through Automation

Texas A&M University

Presenters



Oscar Reyes

@odra94



Andrew Barnett

@andrewmbarnett

Automated User Based Deployment

- This workflow automatically deploys an application to a specific group of people
- The deployment is based on a user group that lives in Microsoft Entra
- The group can only be modified by specific people
- A script takes care of processing the group membership and makes changes to the Jamf server using the Jamf API

Goal

- The goal is to fully automate the deployment of a security tool to people who need to have it
- Aids in automating compliance
- Helps automate a process for our security team
- It reduces time needed to get the tool to the right people through manual scoping

What is the tech stack?

- Microsoft Entra
- Microsoft Entra Graph API
- Jamf server
- Jamf Pro API
- Shell
- Github Actions
- Python (will hopefully be an alternative soon)



What to Build

Enterprise App Registration

- App must have the appropriate API permissions
- You may need to talk to your Microsoft Entra admin
- Test that it works

API / Permissions name	Type	Description
∨ Microsoft Graph (2)		
GroupMember.Read.All	Delegated	Read group memberships
User.ReadBasic.All	Application	Read all users' basic profiles

Entra Security Group

- A group in Microsoft Entra
- This group will be the one read by the Enterprise App we created in the previous slide
- This group will be the source of truth for the deployment
- Group membership can be automated as well

User Extension Attribute

The screenshot shows the Jamf Pro web interface for creating a new user extension attribute. The interface is dark-themed and includes a sidebar on the left with navigation options: Dashboard, Computers, Devices, Users, and Settings. The main content area is titled "New User extension attribute" and contains the following fields:

- Display Name:** A text input field containing "Automated Deployment".
- Description:** A text area containing "This extension attribute will be used to determine which users are marked for the automated deployment workflow."
- Data Type:** A dropdown menu set to "String".
- Input Type:** A dropdown menu set to "Text Field".

At the bottom right of the form, there are "Cancel" and "Save" buttons.

Smart User Group

The screenshot displays the Jamf Pro interface for configuring a Smart User Group. The top navigation bar includes the 'Pro' logo, the user 'Full Jamf Pro', and notification and profile icons. The left sidebar contains a menu with categories: 'Users', 'Inventory', 'Content Management', and 'Groups'. The 'Users' section is expanded, showing 'Smart User Groups' as the active item. The main content area is titled 'Automated Deployment' and includes a breadcrumb 'Users > Smart User Groups'. Below the title are tabs for 'Smart Group', 'Criteria', and 'Reports', with 'Criteria' selected. A checkbox 'Show in Jamf Pro Dashboard' is present. The criteria configuration table is as follows:

AND/OR	CRITERIA	OPERATOR	VALUE
<input type="checkbox"/>	Automated Deployment	is	True

At the bottom right, there are five action buttons: History, View, Clone, Delete, and Edit.

GitHub Repository

The screenshot shows a GitHub repository page for 'EntraUserBasedDeployment' by user 'odra94'. The repository is public and has 7 commits. The main branch is selected. The repository description states: 'This repo is an example that can be used for an automated deployment that is based on a user group made in Microsoft Entra. The workflow takes care of comparing the membership of a smart user group in Jamf Pro against the membership of a security group in Microsoft Entra. If there is a member in the Jamf smart user group that is not in the Entra security group then it will remove that user from the deployment. The Entra security group is the source of truth in this deployment so the smart user group has to match the Entra security group as closely as possible. If there is a member in the Entra security group that is not in the Jamf smart user group, then it will modify the extension attribute of that user and add them to the deployment. Additionally, I have left in the code a lot of standard output to see what is happening as the script is running. Basically a lot of echo commands so you can see what the script is doing. Those can be removed and cleaned up for final deployment. For purposes of this example they have been left in there and I am also just running the commands on the test server. This means only the variables with the prefix JAMP_TEST are the only relevant Jamf related variables being used for this example.'

The repository includes a README file with the following sections:

- Microsoft Entra**: Ensure that you create an enterprise app registration in Microsoft Entra.
- Secrets Needed:** Gather the following information for the secrets you will need for the Microsoft Entra calls:
 - GRAPH_APPLICATION_ID - The Application ID of your app registration
 - GRAPH_CLIENT_SECRET - Create a client secret for your registered app. Grab that secret and store it in the secrets vault for your GitHub repository. Make sure it is a secret and not just a variable.

The repository also shows a file tree with the following files and their commit times:

- github/workflows: Initial commit, 9 minutes ago
- images: Update image size user smart group, 3 minutes ago
- scripts: Initial commit, 9 minutes ago
- .gitignore: Initial commit, 9 minutes ago
- README.md: Cleanup readme, 1 minute ago

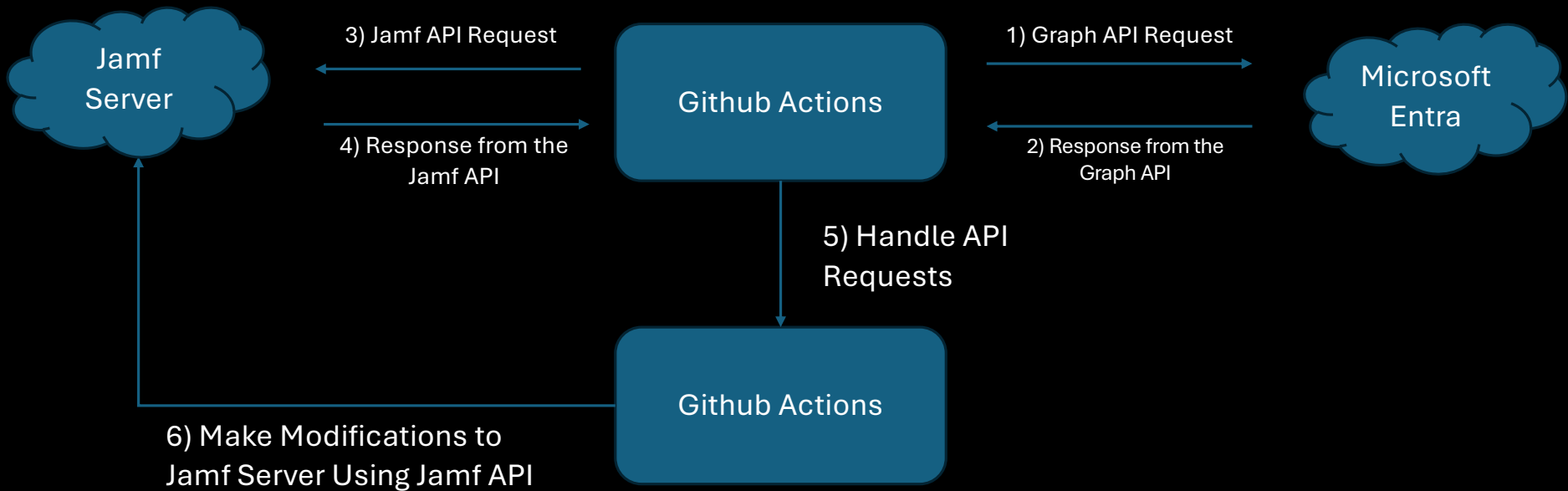
The right sidebar shows the repository's activity, including 0 stars, 1 watching, and 0 forks. There are no releases or packages published.

<https://github.com/odra94/EntraUserBasedDeployment>

Considerations

- Processing only the necessary users
- Reducing the number of API calls that need to happen to reduce load on servers
- Reduce script run time as much as possible
- Keep the workflow scalable and efficient

Workflow



Removing a User From The Deployment

jamf_output.txt

```
johndoe@email.com  
mike@email.com  
jimothy@email.com  
bob@email.com
```

Compare Files

graph_output.txt

```
johndoe@email.com  
mike@email.com  
bob@email.com
```

```
> diff jamf_output.txt graph_output.txt  
3d2  
< jimothy@email.com
```

Adding a New User To The Deployment

jamf_output.txt

```
johndoe@email.com  
mike@email.com  
bob@email.com
```

Compare Files

graph_output.txt

```
johndoe@email.com  
mike@email.com  
jimothy@email.com  
bob@email.com
```

```
> diff jamf_output.txt graph_output.txt  
2a3  
> jimothy@email.com
```

What is Next for this Deployment?

- Defining an established interval to run this workflow based on the security team's needs
- Developing a Python version of this script that is compatible with Microsoft Entra (Azure Functions) or AWS (Lambda)
- Automating the population of the Entra security group
- Automating the removal of the application that was deployed once a user is no longer in scope
- More robust error handling

ReEnroll – Automated Re-Enrollment

- This workflow is designed to automate the re-enrollment process of devices into Jamf Pro
- The deployment is based on Jamf smart groups
- The Jamf smart groups are based off management accounts and extension attributes
- The script uses Jamf API to send commands and gather device inventory information

ReEnroll – Goals

- The goal is to automate the process of re-enrolling a device in Jamf Pro
- Verify the enrollment process is complete and communicating with the Jamf server
- Verify if the LAPS enabled account exists and is valid
- Using swiftDialog, keep the end user informed on the process

What is the tech stack?

- Jamf Pro server
- Jamf Pro API
- Shell
- swiftDialog



How ReEnroll Works?

- **Enrollment Phase**

- Gather device information using Jamf API
- Deploy Jamf Management Framework, Enrollment Invitation, or Profiles Renew – Enrollment command

- **Verify Jamf Communications**

- Check JSS connection
- Policy Check-In
- Inventory Update

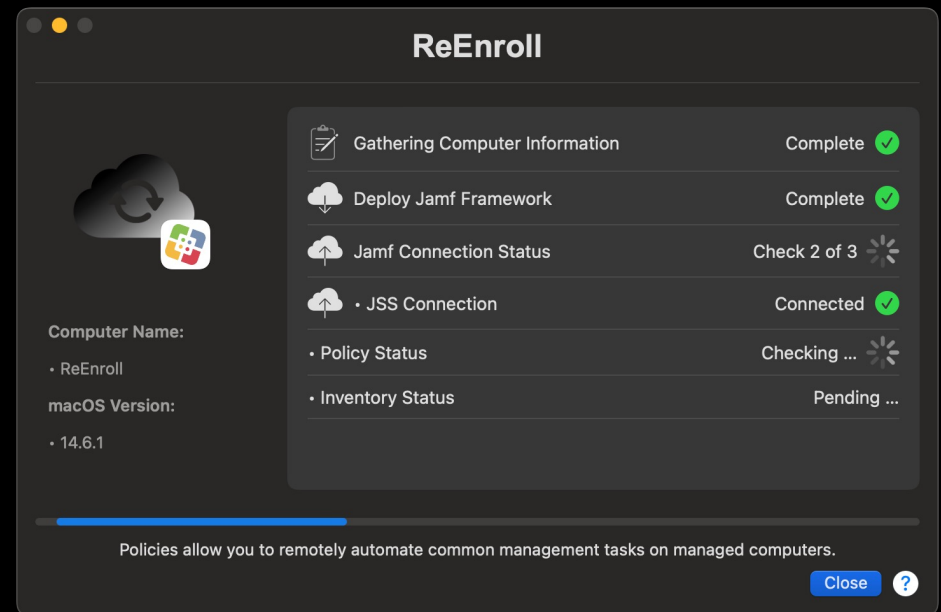
- **Validate Jamf LAPS Account**

- Check for LAPS account
- Verify LAPS password



End-User Experience

- **Interactive**
 - A swiftDialog window will be displayed with progress and information on the workflow
- **Silent**
 - The workflow will run silently in the background to not disturb the end-user





What to Build

Jamf API Account

- Jamf API account must have the appropriate API permissions
- Depending on what workflow you want to accomplish
- You can have more or less API permissions based on your needs

Settings : System > API roles and clients

< ReEnroll Account

Display name Display name for the API Client

ReEnroll Account

API roles Assign roles to determine privileges for the client. Adding multiple roles combines their privileges.

ReEnroll Account

ReEnroll Variables

```
#####  
# Skip Check-In, LAPS Admin Account Username and User to Exempt/Target for Deletion Options  
#####  
  
# Skip Jamf connection verification after enrollment  
skipCheckIN="false" # [ false (default) | true ]  
# Skip LAPS admin account verification after enrollment  
skipLAPSAdminCheck="false" # [ false (default) | true ]  
# LAPS admin account username (add the Jamf managed local admin account username here)  
lapsAdminAccount="$6" # [ add the LAPS admin account username here ]  
# Skip User Exemption/Targeted Deletion  
skipAccountDeletion="false" # [ false (default) | true ]  
# Define the exempt user list from being deleted (Add the username in quotes, with a space in between each)  
exempt_users=("Shared" "Guest" "$loggedInUser") # [ add the exempt user list here ]  
# Define the targeted user list to be deleted (Add the username in quotes, with a space in between each)  
targeted_users=("$lapsAdminAccount" "anotherAccount" ) # [ add the targeted user list here ]  
# Jamf Enrollment Invitation ID (https://company.jamfcloud.com/enroll?invitation=1542270881_!!KwNVnq) (Invitation ID in this example would be: 1542270881)  
enrollmentInvitation="$7" # [ add the Invitation ID here ]  
  
#####  
# ReEnroll Computers Options  
#####  
  
# Send redeploy Jamf Management Framework command  
redeployFramework="true" # [ true (default) | false ]  
# Send enrollment invitation command  
sendEnrollmentInvitation="failure" # [ true | false | failure (default) ]  
# Send profiles renew -type command  
renewProfiles="failure" # [ true | false | failure (default) ]
```


ReEnroll Variables

```
#####  
# Launch Daemon information  
#####  
  
# Skip Launch Daemon after script completion  
skipLaunchDaemon="false"           # [ false (default) | true ]  
# Launch Daemon information  
organizationName="company"         # [ add the organization name here ] #  
organizationReverseDomain="com.company" # [ add the organization reverse domain here ]  
  
#####  
# Additional Settings  
#####  
  
# Add additional logging  
debugMode="false"                  # Debug Mode [ false (default) | verbose ] Verbose adds additional logging  
# Verify and Update Computer Site after enrollment  
updateComputerSite="true"          # Update Computer Site [ true (default) | false ]  
# Move Computer to new Site  
newComputerSiteID=""               # Move Computer Site [ blank (default) ] (Only used if updateComputerSite is true and newComputerSiteName is set)  
  
#####  
# Swift Dialog Settings  
#####  
  
# Display a ReEnroll progress dialog  
displayReEnrollDialog="$8"         # Display ReEnroll Dialog [ true | false (default) ]  
# Unattended Exit Options  
unattendedExit="true"               # Unattended Exit [ true | false (default) ]  
# Unattended Exit Seconds  
unattendedExitSeconds="30"         # Number of seconds to wait until a kill Dialog command is sent
```

ReEnroll - Webhooks



Jamf ReEnroll Yesterday 6:53 PM



ReEnroll: ReEnroll without notification

Computer Name (Serial Number): [REDACTED]

Computer Model: MacBook Pro

Current User: Andrew Barnett [REDACTED]

Notification Status: ReEnroll without notification

ReEnrollment Method: Sending Silent Enrollment Invitation

Computer Record: [REDACTED]

[View in Jamf Pro](#)

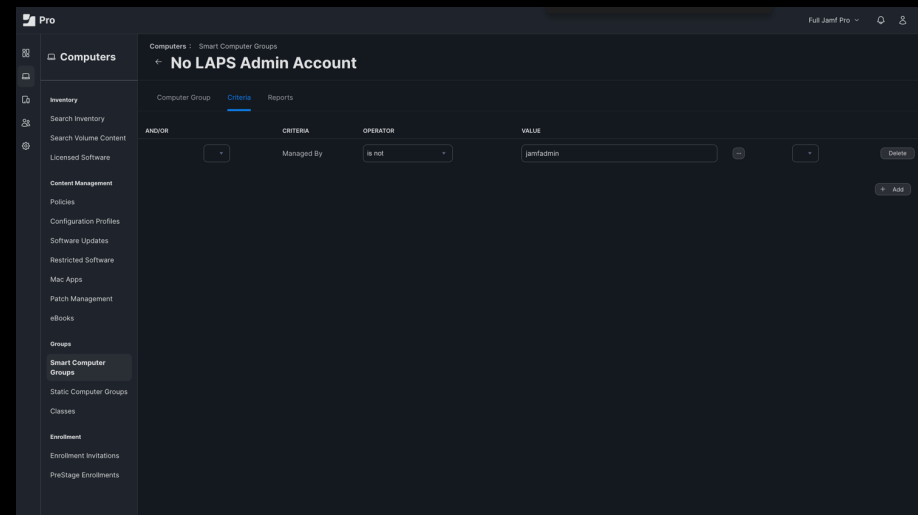
Jamf Extension Attributes

- Jamf Extension attributes to keep inventory updated on workflow
- Can be used to create further Jamf smart groups
- Useful for further troubleshooting of a device if necessary

ReEnroll - Computer Site:	Technology Services
ReEnroll - Property List:	ReEnroll Property List Exists
ReEnroll - ReEnroll Last Run Time:	2025-02-18 18:39:25
ReEnroll - ReEnroll Method:	Enrollment Invitation
ReEnroll - ReEnroll Notification Status:	No Notification
ReEnroll - ReEnroll Profiles Renew Status:	No Renew Profiles Dialog
ReEnroll - ReEnroll Status:	Enrolled Device
ReEnroll - ReEnroll Version:	1.6.1

Jamf Smart Group

- Smart Group Scope
 - Jamf Management Accounts on the device
 - Device last check-in (example: Last Check-In over 30 days)
 - ReEnroll workflow completed



Things To Consider...

- If a device is on macOS 14 or higher and the profiles renew command is sent but didn't go through ADE, the device will get a full screen enrollment prompt
- If you are attempting to add a LAPS enabled account, you will have to use the enrollment invitation or profiles renew command
- If you decide to add a LAPS enabled account, and you have an account being created in PreStage Enrollments. The two accounts can not have the same username.

Future Development

- Manage using Configuration Profiles
- Automation using LaunchDaemon
- Scheduling for automation based on Configuration Profiles

Links to Repositories

- [Automated User Based Deployment](#)
- <https://github.com/AndrewMBarnett/ReEnroll>