CATEGORY: Active Directory

SOP 1 – Get AD User

Script Name: Get AD User **Category:** Active Directory **Version:** 1.0 **Applies To:** RDAM Script Wizard / Script Studio **Approved By:** IT Operations / Security **Last Updated:** \<Set by organization\>

1. Purpose

This script retrieves detailed information about a specific Active Directory user account. It centralizes identity discovery and troubleshooting into a controlled, repeatable workflow, avoiding ad-hoc queries against domain controllers.

2. Scope

- **Systems:** Domain-joined Windows systems with RSAT/AD tools or necessary modules available.
- **Domains:** All trusted AD domains where the operator has read rights.
- **Environments:** Production, test, and development, as allowed by internal policy.
- **Authorized Personnel:**
    - Domain Admins
    - Delegated Helpdesk / IAM operators with read rights to user objects

3. Definitions

- **AD User:** An object of class `user` in Active Directory.
- **Distinguished Name (DN):** The full LDAP path to the AD object.
- **SamAccountName:** Legacy logon name (DOMAIN\user).
- **UPN:** User Principal Name (user@domain.tld).

4. Preconditions

- Operator must have at least **read access** to user objects in the target domain.
- Target domain must be reachable (network/LDAP).
- RDAM Script Wizard must be configured to run under a security context with AD query permissions.
- Any required AD PowerShell module or .NET directory services components must be available.

5. Required Inputs

The Script Wizard will typically prompt for:

- **User Identifier** (at least one of):
    - SamAccountName
    - UPN
    - Distinguished Name (DN)

Optionally:

- **Domain / Domain Controller** scope, if multi-domain / multi-DC targeting is supported.

6. Procedure Steps

1. **Input Collection**
    - The Script Wizard prompts for the user identifier(s).
    - Operator enters the username/UPN/DN as instructed.
    - Wizard validates that at least one identifier is provided.
2. **Input Validation**
    - Check that provided identifier strings are not empty.
    - Optionally validate format of UPN (contains `@`) and DN (contains `=` and `,`).
    - If validation fails, the script aborts with a clear error message and logs the failure.
3. **AD Connection Initialization**
    - Determine the target domain / DC (default or user-specified).
    - Establish connection using the current security context or specified credentials.
    - If connection fails, script logs the issue and returns a failure result.
4. **User Lookup Logic**
    - If DN is provided, query AD directly by DN.
    - Else, if UPN is provided, query by UPN.
    - Else, if SamAccountName is provided, query by SamAccountName (using domain context).
    - If multiple matches are found (e.g., ambiguous names), return a clear warning and either the first match or an error, depending on internal policy.
5. **Attribute Retrieval**
    - Retrieve key attributes such as:

        - `distinguishedName`
        - `samAccountName`
        - `userPrincipalName`
        - `displayName`
        - `mail`
        - `memberOf`
        - `enabled/disabled` status
        - `lastLogonDate`, `pwdLastSet`
    - Format attributes into a structured output (table/JSON/text).
6. **Result Formatting**
    - Present the user data in a readable format (grid/text/JSON) within RDAM.
    - Ensure sensitive attributes (e.g., password-related internal flags) are not exposed beyond policy.
7. **Logging**
    - Log script execution including:
        - Operator ID
        - Timestamp
        - User identifier(s) used
        - Target domain/DC
        - Success/Failure state

7. Expected Output

- Structured information about the requested AD user, including identifiers, status, group memberships (or at least count), and key timestamps.
- Clear indication if the user is **enabled** or **disabled**.
- Clear message if user is **not found** or multiple users match.

8. Post-Execution Validation

- Operator confirms that the returned identity corresponds to the intended user (verify DN/domain).
- If used for troubleshooting, operator may cross-check with another tool (e.g., ADUC) to validate consistency.

9. Error Handling

- **User Not Found:**
    - Script returns “No user found with given identifier” and logs an informational error.
- **Multiple Matches:**
    - Script returns a message indicating ambiguity and may list candidates (depending on configuration).
- **Access Denied:**
    - Script returns an access error and logs it as a permission issue.
- **Connection Errors:**
    - Script returns a connectivity error (e.g., DC unreachable) and logs the DC/Domain attempted.

10. Security Considerations

- Script must only be used by authorized personnel with legitimate business need.
- Output may include sensitive metadata (e.g., group memberships), which can reveal privilege level.
- Results should not be exported or shared outside approved channels.

11. Audit Logging Requirements

- Log at minimum:
    - Operator identity
    - Time of execution
    - Domain/forest targeted
    - User identifier used
    - Outcome (Success/Failure, Not Found, Multiple Matches)

12. Organizational Benefit Statement

This script provides a controlled, auditable way to retrieve AD user details, reducing ad-hoc querying, simplifying troubleshooting, and supporting identity governance while maintaining clear accountability for each lookup.

SOP 2 – Remove User from AD Group

Script Name: Remove User from AD Group **Category:** Active Directory **Version:** 1.0

1. Purpose

This script removes a specified user from a specified Active Directory security or distribution group. It provides a controlled and logged way to de-provision access instead of using manual, error-prone ADUC operations.

2. Scope

- **Systems:** Domain-joined administrative workstations or management servers.
- **Domains:** Any trusted AD domain where group membership changes are authorized.
- **Environments:** Production and lower environments as per internal standards.
- **Authorized Personnel:**
    - IAM staff
    - Domain Admins
    - Delegated group managers with rights to modify group membership

3. Definitions

- **AD Group:** An object of class `group` in Active Directory.
- **Member:** A user, computer, or group listed in the group's `member` attribute.

4. Preconditions

- Operator must have **Write** permission to modify the target group's membership.
- Target domain/DC reachable.
- User and group must exist in AD.
- Change should align with an approved access request, ticket, or de-provisioning process.

5. Required Inputs

Script Wizard prompts for:

- **User identifier:** SamAccountName, UPN, or DN.
- **Group identifier:** SamAccountName, CN, or DN.
- Optional:
    - **Domain/Server** if multi-domain.
    - **Dry Run** flag (if implemented) for validation only.

6. Procedure Steps

1. **Input Collection & Validation**

- Wizard collects user and group identifiers.
- Validates that neither value is empty.
- Optionally validates format.

2. **Resolve User Object**
    - Query AD for the user using provided identifier.
    - If user not found, abort and log.
    - If multiple matches, abort or require refinement.
3. **Resolve Group Object**
    - Query AD for the group.
    - If group not found, abort and log.
4. **Membership Check**
    - Check if user DN is present in group's `member` attribute.
    - If not a member, return an informational message: user not currently in group.
5. **Optional Dry Run Handling**
    - If Dry Run is enabled, show what would happen (user would be removed) without modifying AD.
    - Log that a dry run was performed.
6. **Membership Removal**
    - Remove user from group using AD cmdlet/API (e.g., `Remove-ADGroupMember`).
    - Handle confirmation prompts programmatically (no interactive prompts in production automation).
7. **Post-Change Verification**
    - Re-query the group membership to ensure user is no longer a member.
    - If removal did not succeed, mark as failure and log details.
8. **Logging & Return**
    - Log operation including user, group, operator, time, and result.
    - Return a clear success/failure message to the operator.

7. Expected Output

- On success: confirmation that the user was removed from the specified group.
- On failure: error message explaining why (not found, not a member, permissions, etc.).

8. Post-Execution Validation

- Operator may confirm in ADUC or another tool that the user is no longer associated with the group.
- For privileged groups (e.g., Domain Admins), secondary verification may be required by policy.

9. Error Handling

- **User Not Found:** Abort and log.
- **Group Not Found:** Abort and log.
- **User Not Member:** Log and return informational message (no modification).
- **Access Denied:** Return and log permission error with group DN.
- **Replication Delay:** If user still appears temporarily due to replication, document in logs; operator may recheck later.

10. Security Considerations

- Removing a user from critical groups affects access; ensure action aligns with approved request.
- Script should not be used for bulk changes without additional approvals.
- All membership changes must be logged for compliance and possible forensic review.

11. Audit Logging Requirements

- Operator identity
- Ticket/request reference (if captured via parameter or comment)
- User and group identifiers
- Before/after membership state if possible
- Timestamp and success/failure

12. Organizational Benefit Statement

This script provides a structured, auditable way to revoke group-based access, supporting least privilege, incident response, and regulatory compliance while minimizing the risk of human error in group management.

SOP 3 – Get AD Computer Details

Script Name: Get AD Computer Details **Category:** Active Directory **Version:** 1.0

1. Purpose

This script retrieves detailed information about an Active Directory computer object, used for troubleshooting, inventory, and validation of joined systems.

2. Scope

- Systems: Admin workstations or management servers with AD tools.
- Domains: Any trusted AD domain with computer objects.
- Environments: Production, test, dev.
- Authorized Personnel:
    - Domain Admins
    - Helpdesk/desktop support with delegated read rights

3. Definitions

- **AD Computer:** AD object representing a domain-joined machine.
- **Computer Account:** The credential/identity used for machine authentication.

4. Preconditions

- Operator has read access to computer objects.
- Network connectivity to a DC.
- RDAM able to query AD from current context.

5. Required Inputs

- **Computer Identifier:**
    - SamAccountName (typically `COMPUTER$`)
    - Computer name (NetBIOS or FQDN)
    - DN of the computer object

6. Procedure Steps

1. **Input Collection**
    - Wizard prompts for computer name/identifier.
    - Validate non-empty input.
2. **Resolve Computer Object**
    - Use AD query to find the computer object.
    - Handle ambiguous results (multiple matches) as error or list.

3. **Retrieve Attributes**
    - Gather key attributes:
        - `distinguishedName`
        - `dNSHostName`
        - `operatingSystem`, `operatingSystemVersion`
        - `lastLogonDate`
        - Enabled/disabled state
        - OU location
4. **Format Results**
    - Present in structured output for operators.
5. **Logging**
    - Log who queried which computer and when.

7. Expected Output

- Detailed information about the AD computer object, including OS, last logon, and OU.

8. Post-Execution Validation

- Operator can cross-check in ADUC or monitoring tools if necessary.

9. Error Handling

- **Computer Not Found:** Return and log.
- **Multiple Matches:** Return message with possible matches.
- **Access Issues:** Return and log as permission problem.

10. Security Considerations

- Reveals information about internal systems; access to this script should be limited to authorized support/IT staff.

11. Audit Logging Requirements

- Operator ID
- Computer identifier used
- Timestamp
- Outcome

12. Organizational Benefit Statement

This script standardizes how computer details are retrieved, reducing manual investigation time and supporting asset management, troubleshooting, and security reviews.

SOP 4 – Enable AD Computer Account

Script Name: Enable AD Computer Account **Category:** Active Directory

1. Purpose

This script enables a disabled AD computer account, typically used when reactivating a machine that was decommissioned or disabled during troubleshooting.

2. Scope

- Applies to AD computer objects only.
- Executed by authorized admins or support staff with rights to modify computer accounts.

3. Definitions

- **Disabled Computer Account:** An AD computer object with `userAccountControl` flag set to disabled.

4. Preconditions

- Computer account exists in AD.
- Operator has Write permissions to modify account.
- Action aligns with a valid business request (e.g., ticket).

5. Required Inputs

- Computer name / DN / SamAccountName.

6. Procedure Steps

1. **Input Collection & Validation**
    - Wizard prompts for computer identifier.
    - Validates non-empty.
2. **Resolve Computer Object**
    - Query AD and locate the computer.
    - If not found, abort and log.
3. **Check Current State**

    - Verify if computer is disabled.
    - If already enabled, return informational message.
4. **Enable Operation**
    - Use AD API/cmdlet (e.g., `Enable-ADAccount`) for the computer object.
5. **Post-Change Verification**
    - Requery to ensure account is now enabled.
6. **Logging**
    - Log operator, computer, before/after states, and timestamp.

7. Expected Output

- Confirmation that computer account is enabled, or message that it was already enabled.

8. Post-Execution Validation

- Operator can validate that the machine can now authenticate (e.g., through domain logon or test session).

9. Error Handling

- Not found, access denied, replication delays — logged and reported in a clear message.

10. Security Considerations

- Enabling old or unknown computer accounts may re-introduce risk; ensure the device is trusted and properly managed.

11. Audit Logging Requirements

- Operator ID
- Computer identity
- Previous state and new state
- Approval reference if available

12. Organizational Benefit Statement

This script provides a repeatable, auditable mechanism to re-enable computer accounts, reducing support effort and ensuring all such changes are tracked and justifiable.

SOP 5 – Disable AD Computer Account

Script Name: Disable AD Computer Account **Category:** Active Directory

1. Purpose

This script disables an AD computer account to prevent a device from authenticating to the domain, typically used for decommissioning or isolating compromised machines.

2. Scope

- AD computer objects in any managed domain.
- Executed by authorized admins or incident responders.

3. Definitions

- **Disabled Account:** Account that cannot authenticate against domain controllers.

4. Preconditions

- Computer account exists.
- Operator has Write permissions on the object.
- Action aligns with decommissioning, incident response, or security policy.

5. Required Inputs

- Computer identifier (name, DN, or SamAccountName).

6. Procedure Steps

1. **Input Validation**
    - Collect and validate identifier.
2. **Resolve Computer Object**
    - Query AD.
    - Handle not found or multiple matches.
3. **Check Current State**
    - If already disabled, return info message.
4. **Disable Operation**
    - Use AD cmdlet/API to disable the account.
5. **Post-Change Verification**
    - Requery and confirm disabled flag is set.
6. **Logging**
    - Log who disabled what, when, and why (if reason parameter is used).

7. Expected Output

- Confirmation of successful disable or appropriate error/info.

8. Post-Execution Validation

- Incident or operations staff may try to log on or use tools that confirm authentication failure for that machine.

9. Error Handling

- Not found, access denied, replication issues logged and visible to operator.

10. Security Considerations

- Disabling the wrong machine can disrupt services; use with care and confirm identity.
- For compromised hosts, disabling is usually part of a larger incident workflow.

11. Audit Logging Requirements

- Operator ID
- Computer identity
- Reason (if captured)
- Timestamp and result

12. Organizational Benefit Statement

This script supports secure decommissioning and incident containment by providing a controlled, auditable way to disable machine accounts, reducing risk of unauthorized access.

SOP 6 – Get OU Details

Script Name: Get OU Details **Category:** Active Directory

1. Purpose

This script retrieves information about a specified Organizational Unit (OU), including its DN, parent, and possibly contained object counts, to support design, audit, and troubleshooting work.

2. Scope

- AD OUs across managed domains.
- Read-only operation.

3. Definitions

- **OU:** Container in AD used to organize objects and apply GPOs.

4. Preconditions

- Operator has read access to the OU.
- OU exists in the directory.

5. Required Inputs

- OU DN, or a search path/partial name combined with a domain/context.

6. Procedure Steps

1. **Input Collection**
    - Prompt for OU DN or name/context.
2. **OU Resolution**
    - Query AD to find the OU.
    - Handle multiple matches or not found.
3. **Retrieve Attributes**
    - `distinguishedName`, `name`, parent DN.
    - Optionally number of child objects, linked GPOs, etc.
4. **Formatting Output**
    - Present OU details in a readable, structured format.
5. **Logging**
    - Log which OU was queried, by whom, and when.

7. Expected Output

- OU DN, name, basic metadata, and optionally counts of child objects.

8. Post-Execution Validation

- Operator cross-checks in ADUC if necessary.

9. Error Handling

- Not found, access denied, ambiguous search — clearly returned and logged.

10. Security Considerations

- OU structure can reveal domain design; restrict access to authorized staff.

11. Audit Logging Requirements

- Operator ID

- OU identifier
- Timestamp
- Result

12. Organizational Benefit Statement

This script enables consistent, auditable inspection of OU configuration, supporting better directory design, troubleshooting, and documentation.

SOP 7 – Create New OU

Script Name: Create New OU **Category:** Active Directory

1. Purpose

This script creates a new Organizational Unit in a specified location in the directory, following organizational design and naming standards.

2. Scope

- Used for environment expansion, segmentation, or restructuring.
- Must follow internal OU design standards and approval.

3. Definitions

- **Target DN:** Parent container under which new OU will be created.

4. Preconditions

- Operator has rights to create OUs in the specified parent container.
- OU name has been approved and validated against naming convention.
- Action aligns with architecture/design standards and possibly change management.

5. Required Inputs

- **Parent OU DN or container DN**.
- **New OU Name** (e.g., “Workstations-NYC”).
- Optional: **Protect from accidental deletion** flag.

6. Procedure Steps

1. **Input Collection & Validation**
    - Wizard collects parent DN and OU name.
    - Validate name not empty and follows naming policy.

2. **Parent Container Verification**
    - Query AD to ensure parent OU/container exists.
3. **Uniqueness Check**
    - Check if an OU with same name already exists under the parent.
    - If so, abort to avoid duplicate naming.
4. **Create OU**
    - Execute creation via AD API/cmdlet.
    - Set flags like “Protect from accidental deletion” if required.
5. **Post-Creation Verification**
    - Requery AD to confirm new OU exists at expected DN.
6. **Logging**
    - Log parent DN, new OU DN, operator, and timestamp.

7. Expected Output

- Confirmation including the DN of the newly created OU.

8. Post-Execution Validation

- Administrator can see and manage OU in ADUC.
- GPOs and delegation can then be applied according to separate procedures.

9. Error Handling

- Parent not found, name conflict, access denied are all explicitly returned and logged.

10. Security Considerations

- Creation of OUs impacts delegation and GPO application boundaries; only authorized architects/admins should run this script.

11. Audit Logging Requirements

- Operator ID
- Parent DN
- New OU name and DN
- Reason/ticket if captured

12. Organizational Benefit Statement

This script formalizes OU creation into a controlled, auditable process aligned with directory design, minimizing configuration drift and accidental misplacement of objects.

SOP 8 – Move Object to OU

Script Name: Move Object to OU **Category:** Active Directory

1. Purpose

This script moves an AD object (user, computer, group, etc.) into a specified OU, usually to apply correct policies, delegation, or lifecycle placement.

2. Scope

- Can affect any movable object class in AD.
- Used during onboarding, cleanup, or restructuring.

3. Definitions

- **Source Object:** Object identified by current DN or other identifier.
- **Target OU:** Destination OU DN.

4. Preconditions

- Operator has rights to move the object and to create objects in target OU.
- Target OU exists and is correct per design.

5. Required Inputs

- Object identifier (DN or other key).
- Target OU DN.

6. Procedure Steps

1. **Input Collection & Validation**
    - Collect object identifier and target OU DN.
    - Validate both non-empty.
2. **Resolve Object**
    - Query AD to retrieve current DN.
    - If not found, abort.
3. **Resolve Target OU**

- Verify OU exists.

4. **Pre-Move Checks**
    - Confirm object type is allowed to be moved.
    - Optional: confirm that move aligns with internal rules (e.g., no moving DCs with this tool).
5. **Perform Move**
    - Use AD move operation to update the object's DN to target OU.
6. **Post-Move Verification**
    - Requery object to confirm new DN includes target OU path.
7. **Logging**
    - Log old DN, new DN, operator, and timestamp.

7. Expected Output

- Confirmation that object was moved, showing old and new DN.

8. Post-Execution Validation

- Operator can use ADUC or RDAM tools to confirm the new location and GPO application.

9. Error Handling

- Not found, OU not found, access denied, restricted object classes all handled and logged.

10. Security Considerations

- Moving objects can change applied GPOs and delegation; ensure moves are authorized and traceable.

11. Audit Logging Requirements

- Operator ID
- Object DN before and after
- Target OU
- Timestamp

12. Organizational Benefit Statement

This script ensures object placement changes occur in a controlled, repeatable, and auditable way, reducing misplacement and ensuring correct policy application.

SOP 9 – Get GPO Details

Script Name: Get GPO Details **Category:** Active Directory

1. Purpose

This script retrieves details about a Group Policy Object (GPO), including its GUID, links, and status, for troubleshooting, documentation, and review.

2. Scope

- All GPOs accessible to the operator in the domain.

3. Definitions

- **GPO:** Group Policy Object used to manage configuration on AD-joined machines and users.

4. Preconditions

- Operator has read access to GPOs.
- GPO exists.

5. Required Inputs

- GPO Name, GUID, or other unique identifier.

6. Procedure Steps

1. **Input Collection**
    - Wizard prompts for GPO name or GUID.
2. **Resolution**
    - Query AD/GPMC to identify the correct GPO.
3. **Retrieve Data**
    - Collect properties such as:
        - Display Name
        - GUID
        - Creation/modification time
        - Status (enabled/disabled for user/computer)
        - Links to OUs/sites/domains
4. **Formatting**
    - Present details in structured output.

5. **Logging**
    - Log which GPO was queried and by whom.

7. Expected Output

- Human-readable summary of GPO properties.

8. Post-Execution Validation

- Operator cross-checks with GPMC if necessary.

9. Error Handling

- Not found or access denied errors clearly reported and logged.

10. Security Considerations

- GPO data may reveal security posture; restrict use to authorized admins.

11. Audit Logging Requirements

- Operator ID
- GPO identifier
- Timestamp
- Result

12. Organizational Benefit Statement

This script standardizes GPO inspection, enabling faster troubleshooting and better documentation of policy configuration and linkage.

SOP 10 – Backup GPO

Script Name: Backup GPO **Category:** Active Directory

1. Purpose

This script creates a backup of a specified GPO to a designated file system location for recovery, versioning, and change management.

2. Scope

- Single-GPO backup per execution.
- Target backup path must be accessible and secure.

3. Definitions

- **GPO Backup:** Set of files representing settings of a GPO at a point in time.

4. Preconditions

- Operator has read access to GPO and write access to backup location.
- Backup location complies with organizational backup/retention policies.

5. Required Inputs

- GPO name or GUID.
- Backup folder path.

6. Procedure Steps

1. **Input Collection & Validation**
    - Collect GPO identifier and backup path.
    - Validate path exists or can be created.
2. **Resolve GPO**
    - Confirm GPO exists.
3. **Create Backup**
    - Use GPO backup mechanism (e.g., `Backup-GPO`) to target path.
4. **Post-Backup Verification**
    - Check that backup files exist and operation returned success.
5. **Logging**
    - Log GPO name/GUID, backup location, and operator.

7. Expected Output

- Confirmation that the GPO was backed up to the specified path.

8. Post-Execution Validation

- Operator can verify backup presence and attempt test restore in non-production if required.

9. Error Handling

- GPO not found, path invalid, access denied — all reported and logged.

10. Security Considerations

- Backup files can reveal security and configuration details; must be stored securely with proper access controls.

11. Audit Logging Requirements

- Operator ID
- GPO identifier
- Backup path
- Timestamp

12. Organizational Benefit Statement

This script enforces disciplined, auditable GPO backup practices, supporting safe change management and rapid rollback when needed.

SOP 11 – Link GPO to OU

Script Name: Link GPO to OU **Category:** Active Directory

1. Purpose

This script links an existing GPO to a specified OU to apply its settings to users/computers within that OU.

2. Scope

- Operates on existing GPOs and OUs.
- Used in alignment with GPO design standards.

3. Definitions

- **Link:** Association between a GPO and an AD container (OU/site/domain).

4. Preconditions

- GPO and OU must both exist.
- Operator has rights to manage GPO links on that OU.
- Linking must comply with policy and change management.

5. Required Inputs

- GPO identifier (name/GUID).
- Target OU DN.

- Optional: Link order, enforced flag (depending on implementation).

6. Procedure Steps

1. **Input Collection**
    - Gather GPO identifier and OU DN.
2. **Validation & Resolution**
    - Verify both GPO and OU exist.
3. **Existing Links Check**
    - Check if GPO is already linked to the OU.
    - If already linked, return informational message (optionally adjust order).
4. **Create Link**
    - Use appropriate API/cmdlet to create GPO link to OU.
    - Optionally set link order/enforced.
5. **Post-Link Verification**
    - Re-query OU's GPO links and confirm presence.
6. **Logging**
    - Log GPO, OU, operator, and the nature of the link.

7. Expected Output

- Confirmation that GPO has been linked to the OU, with link properties.

8. Post-Execution Validation

- Operator may run a GPResult or similar tool from a target machine to confirm GPO application after replication.

9. Error Handling

- GPO/OU not found, access denied, conflicting configuration — all clearly reported and logged.

10. Security Considerations

- Linking GPOs can drastically change security posture; only authorized personnel with approvals should perform this action.

11. Audit Logging Requirements

- Operator ID
- GPO identifier

- OU DN
- Timestamp
- Whether the GPO was newly linked or already present

12. Organizational Benefit Statement

This script formalizes and documents GPO linking, ensuring that powerful configuration changes are carried out in a controlled, auditable manner, reducing risk of misconfiguration.

If you’re good with this structure and detail level, next I’ll generate the **Active Directory Advanced** category SOPs in one long message, then proceed category by category until your entire Script Wizard is covered.