==

CATEGORY: Active Directory Advanced

==

SOP 1 – Find Users with Large Token Size

Script Name: Find Users with Large Token Size **Category:** Active Directory Advanced **Version:** 1.0 **Applies To:** RDAM Script Wizard / Script Studio **Approved By:** IT Operations / Security **Last Updated:** \<Set by organization\>

1. Purpose

This script identifies Active Directory user accounts whose Kerberos token size is approaching or exceeding recommended limits due to excessive group membership. Large tokens can cause authentication failures, slow logons, and application access issues. This SOP ensures a controlled, auditable method for detecting such accounts.

2. Scope

- **Systems:** Domain-joined administrative workstations or management servers.
- **Domains:** All trusted AD domains.
- **Environments:** Production, test, and development.
- **Authorized Personnel:**
    - IAM engineers
    - Domain Admins
    - Security analysts
    - Tier-2/3 support

3. Definitions

- **Token Size:** The Kerberos authorization data size generated during logon.
- **Large Token User:** A user whose group membership count or SID history causes token size to exceed Microsoft recommendations (typically > 12 KB).

- **Nested Groups:** Groups inside groups, contributing to token bloat.

4. Preconditions

- Operator must have read access to user and group objects.
- Domain controllers must be reachable.
- AD PowerShell module or equivalent APIs must be available.
- Script Wizard must run under a context with directory query permissions.

5. Required Inputs

- **Target Scope:**
  - Entire domain
  - Specific OU
  - Specific user (optional)
- **Threshold:**
  - Token size threshold (default: 10,000–12,000 bytes)

6. Procedure Steps

1. **Input Collection**
   - Wizard prompts for scope (domain/OU/user) and threshold.
   - Validate threshold is numeric.
2. **Scope Resolution**
   - If OU provided, resolve OU DN.
   - If user provided, resolve user DN.
   - If domain-wide, enumerate all users.
3. **Group Membership Enumeration**
   - For each user:
     - Retrieve direct group memberships.
     - Expand nested groups recursively.
     - Count total SIDs.
4. **Token Size Calculation**
   - Estimate token size using Microsoft’s documented formula.
   - Compare against threshold.

5. **Result Filtering**
    - Identify users exceeding or approaching threshold.
    - Sort results by token size descending.
6. **Output Formatting**
    - Present results in table/JSON format.
    - Include:
        - User
        - Total groups
        - Estimated token size
        - OU
        - SID history count
7. **Logging**
    - Log scope, threshold, operator, and number of users flagged.

7. Expected Output

- A list of users with large or borderline token sizes.
- Clear indicators of risk level.

8. Post-Execution Validation

- IAM team may review group memberships for flagged users.
- Security team may validate impact on authentication.

9. Error Handling

- **Access Denied:** Log and abort.
- **User/OU Not Found:** Return clear error.
- **DC Unreachable:** Log connectivity failure.

10. Security Considerations

- Group membership data is sensitive; restrict access.
- Results may reveal privileged accounts.

11. Audit Logging Requirements

- Operator ID

- Scope
- Threshold
- Number of users flagged
- Timestamp

12. Organizational Benefit Statement

This script proactively identifies identity risks and authentication bottlenecks, reducing outages and improving security posture by preventing token bloat issues.

SOP 2 – Report Group Nesting Depth

Script Name: Report Group Nesting Depth **Category:** Active Directory Advanced

1. Purpose

This script analyzes the depth and complexity of nested AD group structures. Excessive nesting can cause token bloat, slow logons, and unpredictable access inheritance. This SOP ensures consistent, auditable analysis of group hierarchy.

2. Scope

- AD security and distribution groups.
- All trusted domains.
- Used by IAM, security, and architecture teams.

3. Definitions

- **Nesting Depth:** Number of layers of groups inside groups.
- **Recursive Membership:** Membership inherited through nested groups.

4. Preconditions

- Operator must have read access to group objects.
- AD module available.
- Domain reachable.

5. Required Inputs

- Group identifier (name, SamAccountName, or DN).
- Optional: Maximum recursion depth.

6. Procedure Steps

1. **Input Collection**
    - Wizard prompts for group identifier.
2. **Group Resolution**
    - Query AD to locate group.
    - Abort if not found.
3. **Recursive Enumeration**
    - Traverse nested groups.
    - Track depth, cycles, and repeated references.
4. **Cycle Detection**
    - Detect circular nesting (rare but possible).
    - Flag cycles in output.
5. **Depth Calculation**
    - Determine maximum nesting depth.
    - Count total nested groups.
6. **Output Formatting**
    - Provide tree-style or table output.
    - Include warnings for deep nesting.
7. **Logging**
    - Log group, depth, operator, timestamp.

7. Expected Output

- Maximum nesting depth.
- List of nested groups.
- Cycle warnings if applicable.

8. Post-Execution Validation

- IAM may review group design for simplification.
- Security may evaluate privilege inheritance.

9. Error Handling

- Group not found.
- Access denied.
- Recursion limit exceeded.

10. Security Considerations

- Group structures reveal privilege architecture; restrict access.

11. Audit Logging Requirements

- Operator ID
- Group identifier
- Depth result
- Timestamp

12. Organizational Benefit Statement

This script improves identity hygiene by exposing overly complex group structures that can lead to privilege escalation or authentication failures.

SOP 3 – Enumerate SID History

Script Name: Enumerate SID History **Category:** Active Directory Advanced

1. Purpose

This script retrieves the SID History attribute for AD objects, typically used during migrations or forensic investigations. SID History can indicate legacy access or unauthorized privilege inheritance.

2. Scope

- AD users, groups, and computers.
- Used by IAM, security, and migration teams.

3. Definitions

- **SID History:** Legacy SIDs preserved during migrations for backward compatibility.
- **Security Identifier (SID):** Unique identifier for security principals.

4. Preconditions

- Operator must have read access to SIDHistory attribute.

- Object must exist.

5. Required Inputs

- Object identifier (DN, SamAccountName, UPN).

6. Procedure Steps

1. **Input Collection**
    - Wizard prompts for object identifier.
2. **Object Resolution**
    - Query AD to locate object.
3. **Retrieve SID History**
    - Read `SIDHistory` attribute.
    - If empty, return “No SID History present.”
4. **Format Output**
    - List SIDs with timestamps if available.
5. **Logging**
    - Log object and operator.

7. Expected Output

- List of SIDs in SIDHistory.

8. Post-Execution Validation

- IAM may validate whether SIDHistory entries are legitimate.

9. Error Handling

- Object not found.
- Access denied.

10. Security Considerations

- SIDHistory can be abused for privilege escalation; restrict access.

11. Audit Logging Requirements

- Operator ID
- Object identifier

- SIDHistory count
- Timestamp

12. Organizational Benefit Statement

This script supports secure identity management by exposing legacy or suspicious SIDHistory entries that may pose security risks.

SOP 4 – Shadow Group Sync

Script Name: Shadow Group Sync **Category:** Active Directory Advanced

1. Purpose

This script synchronizes membership between a source group and a “shadow” group, ensuring that the shadow group mirrors the source group’s membership. Used for delegated access, application scoping, or privilege separation.

2. Scope

- Security groups only.
- Used by IAM and application teams.

3. Definitions

- **Shadow Group:** A group that mirrors another group’s membership.
- **Source Group:** The authoritative group.

4. Preconditions

- Operator must have read access to source group and write access to shadow group.
- Both groups must exist.

5. Required Inputs

- Source group identifier.
- Shadow group identifier.
- Optional: Remove extraneous members flag.

6. Procedure Steps

1. **Input Collection**
    - Wizard prompts for source and shadow groups.

2. **Resolution**
    - Query AD to locate both groups.
3. **Membership Enumeration**
    - Retrieve source group members.
    - Retrieve shadow group members.
4. **Comparison**
    - Identify missing members in shadow group.
    - Identify extra members in shadow group (if removal enabled).
5. **Synchronization**
    - Add missing members.
    - Remove extra members (if enabled).
6. **Verification**
    - Requery shadow group to confirm match.
7. **Logging**
    - Log adds/removes, operator, timestamp.

7. Expected Output

- Summary of added/removed members.
- Confirmation that shadow group matches source.

8. Post-Execution Validation

- IAM may verify group membership in ADUC.

9. Error Handling

- Group not found.
- Access denied.
- Circular reference (shadow = source).

10. Security Considerations

- Shadow groups may control application access; ensure changes are authorized.

11. Audit Logging Requirements

- Operator ID

- Source and shadow groups
- Adds/removes
- Timestamp

12. Organizational Benefit Statement

This script ensures consistent, controlled group synchronization, reducing manual effort and preventing privilege drift.

SOP 5 – Report AD Object ACL

Script Name: Report AD Object ACL **Category:** Active Directory Advanced

1. Purpose

This script retrieves the Access Control List (ACL) of an AD object, enabling security teams to audit permissions, detect misconfigurations, and validate delegation.

2. Scope

- AD objects of any class.
- Used by IAM, security, and auditors.

3. Definitions

- **ACL:** List of permissions applied to an object.
- **ACE:** Individual permission entry.

4. Preconditions

- Operator must have rights to read object security descriptors.
- Object must exist.

5. Required Inputs

- Object DN or identifier.

6. Procedure Steps

1. **Input Collection**
    - Wizard prompts for object identifier.
2. **Object Resolution**
    - Query AD to locate object.

3. **Retrieve ACL**
    - Read security descriptor.
    - Extract ACEs.
4. **Format Output**
    - Display ACEs with:
        - Trustee
        - Rights
        - Inheritance
        - Type (Allow/Deny)
5. **Logging**
    - Log object and operator.

7. Expected Output

- Full ACL listing.

8. Post-Execution Validation

- Security team may review for excessive permissions.

9. Error Handling

- Access denied.
- Object not found.

10. Security Considerations

- ACLs reveal sensitive delegation; restrict access.

11. Audit Logging Requirements

- Operator ID
- Object DN
- ACE count
- Timestamp

12. Organizational Benefit Statement

This script provides a controlled, auditable method for reviewing AD permissions, supporting least privilege and compliance.

SOP 6 – Find Stale AD Computers

Script Name: Find Stale AD Computers **Category:** Active Directory Advanced

1. Purpose

This script identifies computer accounts that have not authenticated or updated attributes within a defined timeframe, supporting cleanup, security hardening, and lifecycle management.

2. Scope

- All AD computer objects.
- Used by IAM, desktop engineering, and security.

3. Definitions

- **Stale Computer:** A computer account with no recent logon or password update.

4. Preconditions

- Operator must have read access to computer objects.
- Domain reachable.

5. Required Inputs

- **Staleness Threshold:** Days since last logon (e.g., 30/60/90).
- Optional: OU scope.

6. Procedure Steps

1. **Input Collection**
    - Wizard prompts for threshold and optional OU.
2. **Scope Resolution**
    - Enumerate computers in domain or OU.
3. **Attribute Retrieval**
    - Read `lastLogonTimestamp` or equivalent.
4. **Staleness Calculation**
    - Compare timestamp to threshold.
5. **Filtering**
    - Identify computers exceeding threshold.

6. **Output Formatting**
    - Provide list with:
        - Computer name
        - Last logon
        - OU
        - Enabled/disabled state
7. **Logging**
    - Log threshold, count, operator.

7. Expected Output

- List of stale computers.

8. Post-Execution Validation

- Desktop engineering may validate whether machines are truly decommissioned.

9. Error Handling

- Access denied.
- Timestamp missing (rare).

10. Security Considerations

- Stale accounts pose security risk; results should be reviewed promptly.

11. Audit Logging Requirements

- Operator ID
- Threshold
- Count of stale computers
- Timestamp

12. Organizational Benefit Statement

This script supports AD hygiene by identifying unused or abandoned computer accounts, reducing attack surface and improving directory accuracy.