# Implementing Granular Security Controls in CA ACF2®

Dave Klopf, Software Engineer

**August 26, 2016**

# Agenda

1 PRESENTATION OBJECTIVES

2 INTRODUCTION TO THE CASECAUT RESOURCE CLASS

3 CONTROLLING PASSWORD/PASSWORD FIELD ADMINISTRATION

4 EXAMPLE 1:  HELP-DESK ADMINISTRATOR

5 CONTROLLING CERTIFICATE ADMINISTRATION

6 EXAMPLE 2 & 3:  CERTIFICATE ADMINISTRATION

7 RESOURCES, FINAL THOUGHTS, & QUESTIONS

# Objectives

By the end of this presentation, you should understand:

- What the CA ACF2® CASECAUT resource class feature enables.
- How the CA ACF2® CASECAUT resource class feature provides granular security.
- Why it is absolutely necessary to implement granular security controls for mainframe environments.

# Introduction to the CASECAUT Resource Class

- **What is it?**

  A pre-defined resource class (CASECAUT) with internal CLASSMAP record of TYPE=AUT, new to ACF2® r15.

- **What does it do?**

  Supplements existing administrative authorities by providing the ability to authorize users to perform administrative functions over passwords, password fields, and certificates without adding any high-level privileges to the user.

- **Why would I use this?**

  To granularly control administrative functions in order to prevent users from performing administration tasks that they should not be authorized to do.  For instance, modifying the passwords for user ID's outside of their scope, like a high-level security admin.  Conversely, CASECAUT can be used to allow certain administrative functions for use by an ID while blocking others.  For instance, allowing a Help-Desk administrator to modify another user's password, but not change any of the password requirements, like number of special characters required.

# Introduction to the CASECAUT Resource Class

- **How does CASECAUT work in CA ACF2®?**

  CASECAUT provides granular security controls using simple ACF2® resource rules.  Resource entities for these rules are based on the field being changed.  For instance, *ACFCMD.USER.PASSWORD* controls administration over user passwords.

  When being processed, CASECAUT class SAF calls are internally enforced.  Bypassing the security check via SAFDEF override is not allowed.  When traced, SECTRACE output displays "SAFDEF=+ENFORCE" for CASECAUT class events.  CASECAUT rules can have logging set for failures, LOG, and PREVENT rules.  CASECAUT rules follow all Scope controls for the target user to restrict which users they can change.

- **What can I protect with CASECAUT?**

  CASECAUT protects two groups of administrative functions:  *ACFCMD.USER* and *ACFCMD.DIGTCERT*. These two groups correspond to user's password/user's status and certificate related administration commands, respectively.

# Controlling Password/Password Field Administration

- **CASECAUT provides control using the *ACFCMD.USER.fieldname* resource rules.**

  These rules only allow for changes to password or user status related fields; no inserts or deletions of users is allowed.  Thus, protecting against unauthorized users creating or deleting users outside of their scope.

- **Requirements for using *ACFCMD.USER.fieldname* resource rules are:**
  - "Requestor" and "Target" end users must not be SECURITY, ACCOUNT, AUDIT, LEADER, or CONSULT LIDs.
  - No SERVICE permissions allowed on rule lines; only access permissions are allowed.
  - Appropriate SCOPE records exist for the target user(s) to restrict which IDs the target user may change.
  - CASECAUT resource rules must be resident (i.e. add R-RAUT type to GSO INFODIR rec)

# Controlling Password/Password Field Administration

| Field Name | CASECAUT Resource Name |
|---|---|
| PASSWORD | ACFCMD.USER.PASSWORD |
| PWPHRASE | ACFCMD.USER.PWPHRASE |
| PWP-VIO | ACFCMD.USER.PWP-VIO |
| PSWD-VIO | ACFCMD.USER.PSWD-VIO |
| PSEDCVIO | ACFCMD.USER.PSWDCVIO |
| KERB-VIO | ACFCMD.USER.KERB-VIO |
| CANCEL | ACFCMD.USER.CANCEL |
| SUSPEND | ACFCMD.USER.SUSPEND |

# Example 1: Help-Desk Administrator

- **Scenario:**

  You have a set of Help-Desk Administrators who have the role HLPDSK1 and wish to allow them to perform password changes and unsuspend users in their scope. What might a CASECAUT resource rule look like that provides this level of access?

- **Solution:**

```
SET RESOURCE(AUT)
COMP * STORE
$KEY(ACFCMD) TYPE(AUT) ROLESET
  USER.PASSWORD ROLE(HLPDSK1) ALLOW
  USER.SUSPEND ROLE(HLPDSK1) ALLOW
END

F ACF2,REBUILD(AUT),C(R)
```

## Resulting Behavior:

If TSTUSR01 is a user under a HLPDSK1 administrator's scope, the administrator can perform password and suspend administration against TSTUSR01. Conversely, if SECADM is a privileged security administrator outside of a HLPDSK1 administrator's scope, the administrator is unable to perform any password or suspend administration against SECADM.

# Controlling Certificate Administration

- **CASECAUT provides control using the *ACFCMD.DIGTCERT.fieldname* resource rules.**

  These rules only allow for changes to digital certificate and keyring related fields via commands through TSO/E or Batch processing.  Thus, providing users the ability to control certificate administration for themselves and others within their scope.

- **Requirements for using *ACFCMD.DIGTCERT.fieldname* resource rules are:**
  - SERVICE levels control the type of access:
    - SERVICE(READ) – User can administer own certificate, keyring, or token.
    - SERVICE(UPDATE) – User can administer another user's certificate, keyring, or token.
    - SERVICE(DELETE) – User can administer a SITE or CERTAUTH certificate and/or certificate mapping.
  - Appropriate SCOPE records exist for the target user(s) to restrict which user certificates the target user may administer.
  - CASECAUT resource rules must be resident (i.e. add R-RAUT type to GSO INFODIR rec)

# Controlling Certificate Administration

| Certificate Command | Resource Names |
|---|---|
| CHKCERT | ACFCMD.DIGTCERT.CHKCERT |
| CHANGE | ACFCMD.DIGTCERT.ALTER |
| CONNECT | ACFCMD.DIGTCERT.CONNECT |
| DELETE | ACFCMD.DIGTCERT.DELETE |
| EXPORT | ACFCMD.DIGTCERT.EXPORT |
| EXPORT (KEYRING) | ACFCMD.DIGTCERT.EXPORTKEY |
| GENCERT | ACFCMD.DIGTCERT.GENCERT |
| GENREQ | ACFCMD.DIGTCERT.GENREQ |

# Controlling Certificate Administration

| Certificate Command | Resource Names |
|---|---|
| INSERT | ACFCMD.DIGTCERT.ADD |
| INSERT (CERTMAP) | ACFCMD.DIGTCERT.ADDMAP |
| INSERT (KEYRING) | ACFCMD.DIGTCERT.ADDRING |
| LIST | ACFCMD.DIGTCERT.LIST |
| P11TOKEN BIND | ACFCMD.DIGTCERT.P11TOKEN.BIND |
| P11TOKEN IMPORT | ACFCMD.DIGTCERT.P11TOKEN.IMPORT |
| P11TOKEN UNBIND | *No CASECAUT auth's required.* |
| REKEY | ACFCMD.DIGTCERT.REKEY |

# Controlling Certificate Administration

| Certificate Command | Resource Names |
|---|---|
| REMOVE | ACFCMD.DIGTCERT.REMOVE |
| RENEW | ACFCMD.DIGTCERT.RENEW |
| ROLLOVER | ACFCMD.DIGTCERT.ROLLOVER |

# Example 2: Certificate Administration

- **Scenario:**

  You have a set of Help-Desk Administrators who have the role HLPDSK1 and wish to allow them to administer all certificate-related objects for users in their scope and have those administration commands logged by ACF2. What might a CASECAUT resource rule look like that provides this level of access?

- **Solution:**

```
SET RESOURCE(AUT)
COMP * STORE
$KEY(ACFCMD) TYPE(AUT) ROLESET
  DIGTCERT.- ROLE(HLPDSK1) –
  SERVICE(READ,UPDATE,DELETE) LOG
END

F ACF2,REBUILD(AUT),C(R)
```

## Resulting Behavior:

If TSTUSR01 is a user under a HLPDSK1 administrator's scope, the administrator can perform any certificate-related administration against TSTUSR01. Conversely, if SECADM is a privileged security administrator outside of a HLPDSK1 administrator's scope, the administrator is unable to perform any certificate-related administration against SECADM.

# Example 3: Certificate Administration

- **Alternate Scenario:**

  You have a set of Help-Desk Administrators who have the role HLPDSK1 and wish to only allow them to connect and remove certificates to a user’s keyring for all users within their scope. What might a CASECAUT resource rule look like that provides this level of access?

- **Solution:**

```
SET RESOURCE(AUT)
COMP * STORE
$KEY(ACFCMD) TYPE(AUT) ROLESET
  DIGTCERT.CONNECT ROLE(HLPDSK1) –
    SERVICE(UPDATE)
  DIGTCERT.REMOVE ROLE(HLPDSK1) –
    SERVICE(DELETE)
END
F ACF2,REBUILD(AUT),C(R)
```

## Resulting Behavior:

If TSTUSR01 is a user under a HLPDSK1 administrator’s scope, the administrator can only connect or remove certificates from TSTUSR01’s keyring. Conversely, if SECADM is a privileged security administrator outside of a HLPDSK1 administrator’s scope, the administrator is unable to connect or remove certificates from SECADM’s keyring.

# Resources

- More detailed documentation, including command authorization requirements, can be found in the CA ACF2® product documentation on the new Docops Documentation Platform: https://docops.ca.com/ca-acf2-for-z-os/15-0-&-16-0/en
- Specifics to this presentation can be found under the following subheadings within the "Administrating" section:
  - Identify Who Can Maintain Logonid Records
  - Granular Certificate Administration

# Final Thoughts

- **If you like what you saw, join CA Communities and start submitting Ideation requests for future granularity enhancements.**

  We want to continue to provide the product support and enhancements that best suit your business needs. In order to do this, your participation in the CA Communities and Ideation spaces is critical. By submitting Ideation requests, other Communities members can see your ideas and vote on them. With these votes, we at CA gain a better understanding of what is important to you, our customers, and can better plan and act on those enhancements that are the most meaningful and provide the most value to your business.

# Questions?

?

# CA Mainframe Security Monthly Fridays

| Topic | Date |
|---|---|
| Implementing Granular Security Controls in CA ACF2 | August 26 |
| Defense in Depth Enterprise Security: Mainframe Security 101 | September 23 |
| Introducing CA Compliance Event Manager | October 28 |
| CA World ‘16 | November 14 - 18 |
| Real-Time Data Audit and Security | December 23 |

Control

Find

Classify

Audit

Protect

- **Where: CA Mainframe Security Community**
- **When: Every 4th Friday of the month**

ca® technologies

**Dave Klopf**

Software Engineer

David.Klopf@ca.com