

---

# R\_admin – RACF's Administration API

---

Vanguard Security Expo 2008

Session RAA5

Bruce Wells - IBM

[brwells@us.ibm.com](mailto:brwells@us.ibm.com)

---

# Disclaimer

**The information contained in this document is distributed on an "as is" basis, without any warranty either express or implied. The customer is responsible for use of this information and/or implementation of any techniques mentioned. IBM has reviewed the information for accuracy, but there is no guarantee that a customer using the information or techniques will obtain the same or similar results in its own operational environment.**

**In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used. Functionally equivalent programs that do not infringe IBM's intellectual property rights may be used instead. Any performance data contained in this document was determined in a controlled environment and therefore, the results which may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.**

**It is possible that this material may contain references to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM Products, programming or services in your country.**

**IBM retains the title to the copyright in this paper as well as title to the copyright in all underlying works. IBM retains the right to make derivative works and to republish and distribute this paper to whomever it chooses.**

---

---

# Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

- z/OS
- RACF

\* Registered trademarks of IBM Corporation

**The following are trademarks or registered trademarks of other companies.**

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

\* All other products may be trademarks or registered trademarks of their respective companies.

## Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

---

---

# Agenda

- API overview
  - Description of each of the functions
  - Documentation “demo”
-

---

# Overview

- The R\_admin callable service (IRRSEQ00) is an assembler programming interface which allows for management of RACF profiles and system wide settings (SETROPTS)
  - Easier to use than RACROUTE or ICHEINTY
  - Documentation completely rewritten for z/OS V1R7
-

---

## R\_admin functions

- Run a RACF command
    - By providing a command image
    - By providing tokenized data
  - Extract user or group profile information
  - Extract SETROPTS settings
  - Retrieve a PKCS#7 password envelope
-

---

## R\_admin as a SAF Interface

- R\_admin called by SAF router, subject to SAF exits
  - But it is a highly RACF-specific interface
    - Segment names, field names, data format
  - Don't expect this to be a general administrative interface which will work regardless of the underlying security product
-

---

## R\_admin and LDAP

- Consider using LDAP and the SDBM backend in order to retrieve and update RACF data
    - Open, remote-able interface, callable by java, C, and REXX
    - Restricted to users, groups and connections
-



---

# Call parameters

```
CALL IRRSEQ00, (Work_area, /* Common parms */
                ALET, SAF_return_code, /* for all the */
                ALET, RACF_return_code, /* RACF callable */
                ALET, RACF_reason_code, /* services */
                Function_code, /* Requested fcn */
                Parm_list, /* Input p-list */
                RACF_userID, /* "Run-as" user */
                ACEE_ptr, /* "Run-as" ACEE */
                Out_message_subpool, /* Output subpool */
                Out_message_strings /* Output anchor */
                ), VL
```

---

---

## R\_admin General Attributes

- Caller specifies the function to perform and provides a function-specific parameter list
- Caller provides a subpool and address field for the output
- Supervisor state callers can specify an identity under whose authority the request will run
- Some functions are available to problem state callers, and are protected by FACILITY resources
- Most functions require the RACF subsystem address space. Caller does not require a TSO environment.

---

Note: IRRPCOMP macro provides some mappings and constants

---

# Run command



# Run-command

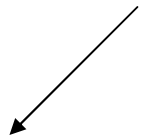
- Caller provides RACF command image as function-specific parameter list
- R\_admin sends command to RACF address space for execution
- Command output is returned to caller
- Amount of output is restricted to 4096 **lines** (not bytes) of output

Function code	Authorization	RACF address space required
5	<ul style="list-style-type: none"><li>■ Command processor authorization</li><li>■ FACILITY - IRR.RADMIN.&lt;cmd-name&gt; (READ)</li></ul>	Yes

---

# Run command – input format

Parm\_list



Length (2 bytes)	Left-justified RACF command image ...
------------------	---------------------------------------

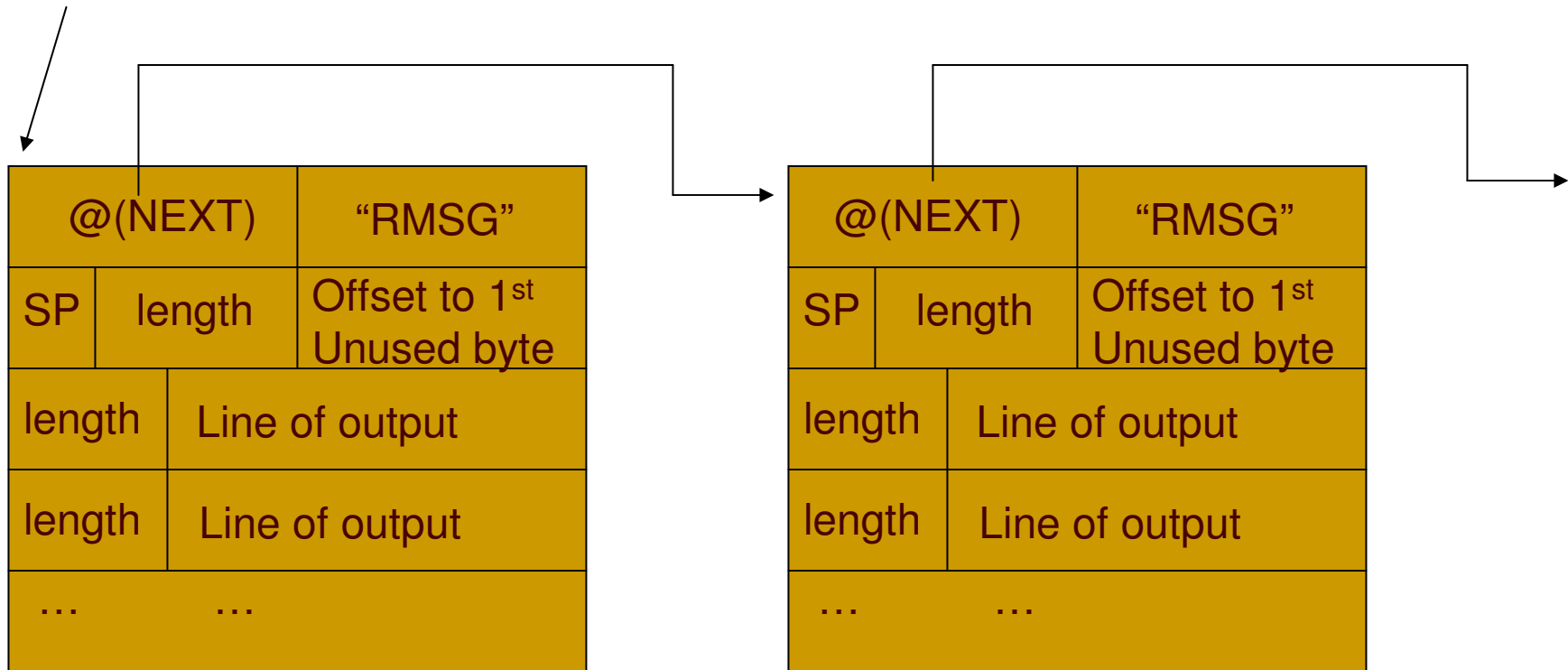
For example

41	ALTUSER GEORGE REVOKE NOSPECIAL NAME(MUD)
----	---

---

# Run command – output format

Out\_message\_strings



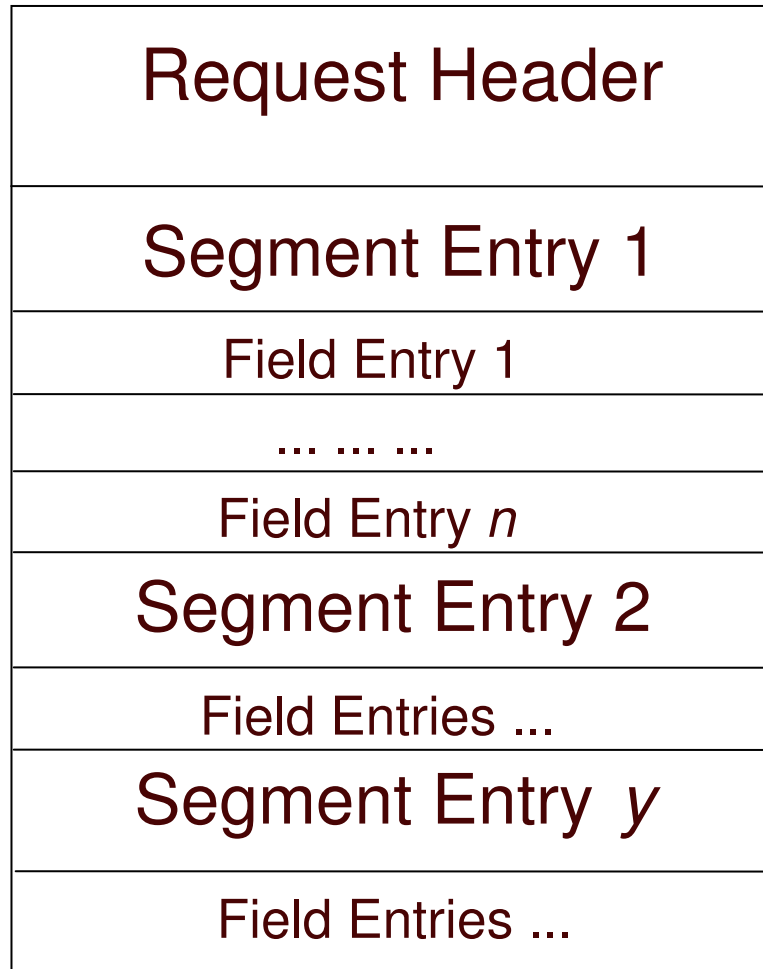
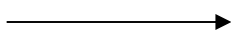
# Run command - tokenized

- Caller provides architected input structure
- Add, alter, delete, and list commands supported for each of the profile types
  - Including CONNECT, REMOVE, PERMIT
- SETROPTS also supported
- R\_admin creates the command image internally from input parameter list
- Command output returned same as for run-cmd

Function codes	Authorization	RACF address space required
1-4, 6-21	<ul style="list-style-type: none"><li>■ Supervisor state</li><li>■ Command processor authorization</li></ul>	Yes

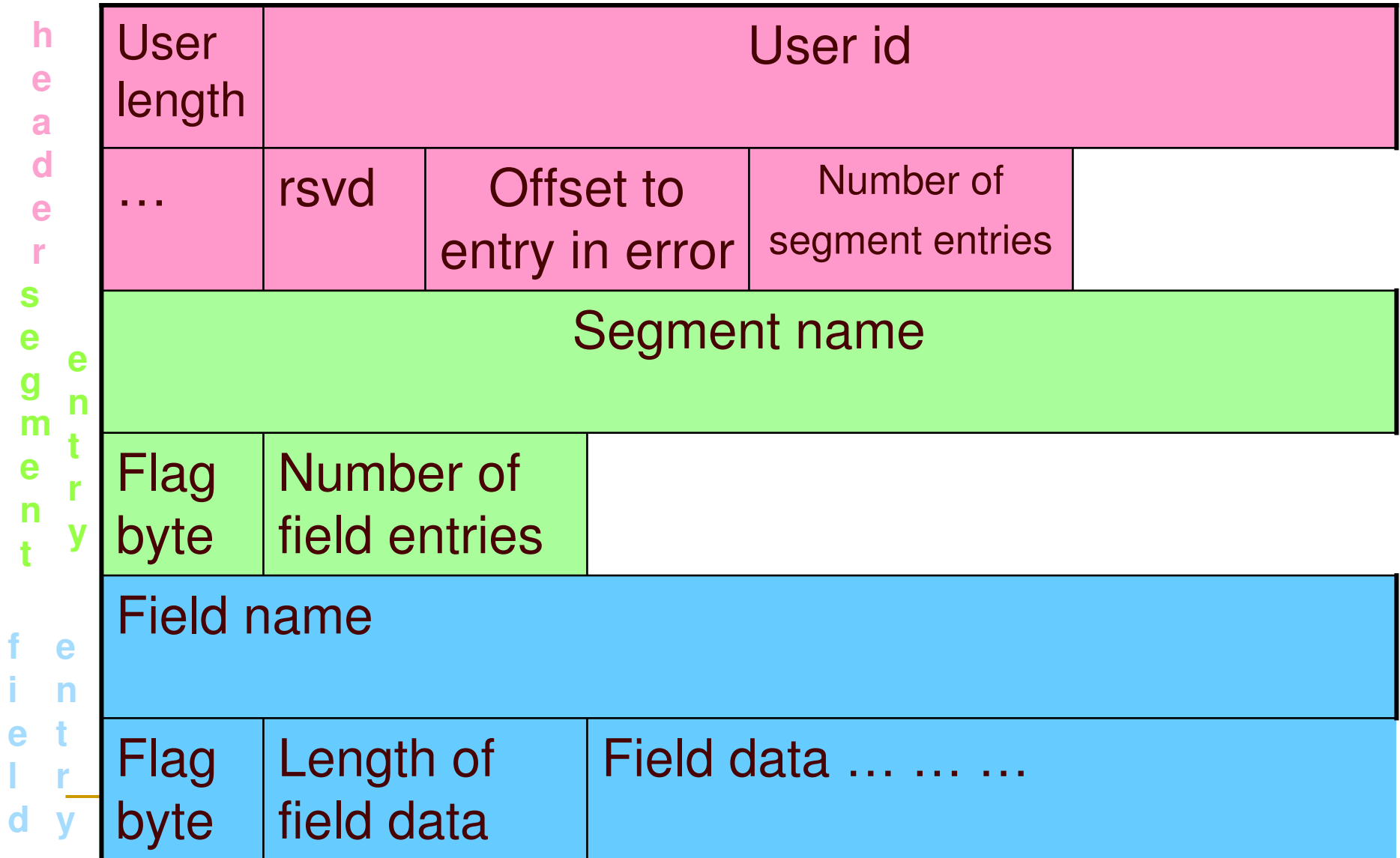
# Tokenized input format

Parm\_list

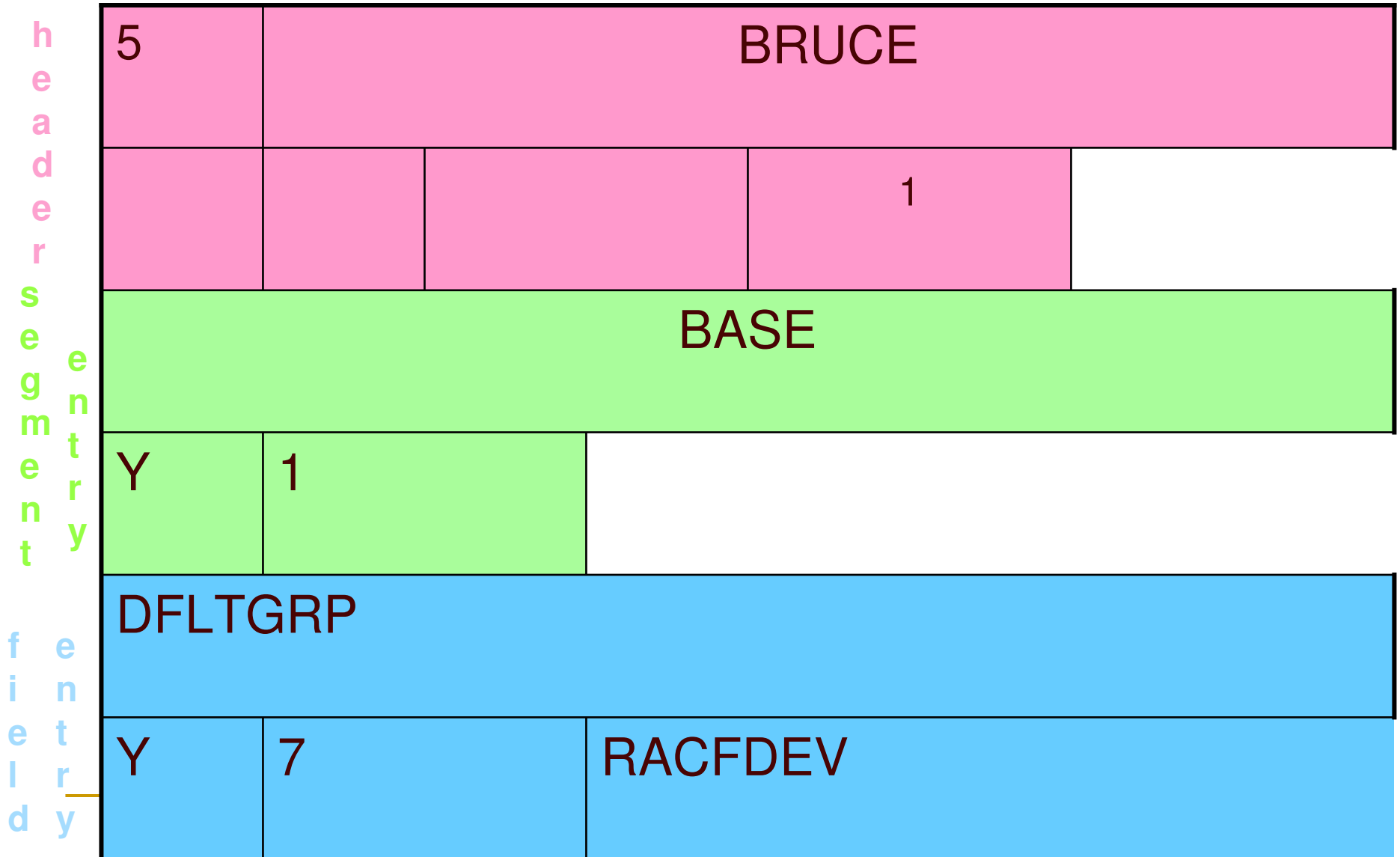




# Detailed mapping for USER request



# Detailed mapping ... example



---

# Code Example: Add user BRUCE

```
HEADER DC AL1(5),CL8'BRUCE',AL1(0),AL2(0),AL2(2)
BSEG   DC CL8'BASE',CL1'Y',AL2(3)
BFLD1  DC CL8'NAME',CL1'Y',AL2(13),CL13''''BRUCE WELLS''''
BFLD2  DC CL8'OWNER',CL1'Y',AL2(7),CL7'RACFDEV'
BFLD3  DC CL8'SPECIAL',CL1'Y',AL2(0)
OSEG   DC CL8'OMVS',CL1'Y',AL2(3)
OFLD1  DC CL8'UID',CL1'Y',AL2(4),CL4'3500'
OFLD2  DC CL8'HOME',CL1'Y',AL2(10),CL10'/u/brwells'
OFLD3  DC CL8'PROGRAM',CL1'Y',AL2(7),CL7'/bin/sh'
```

Is the equivalent of:

```
ADDUSER BRUCE NAME('BRUCE WELLS') OWNER(RACFDEV) SPECIAL
        OMVS(UID(3500) HOME(/u/brwells) PROGRAM(/bin/sh))
```

---

---

# Profile Extract Functions



# Profile extract functions

- Extract User, Group and Connect information from the RACF database in an architected format which is a programming interface
- No limit imposed on output size
- Requires same authority as LISTUSER/GRP
- All (authorized) profile data returned

Function codes	Authorization	RACF address space required
25-29	<ul style="list-style-type: none"><li>■ Command processor authorization</li><li>■ FACILITY - IRR.RADMIN.&lt;cmd-name&gt; (READ)</li></ul>	No

---

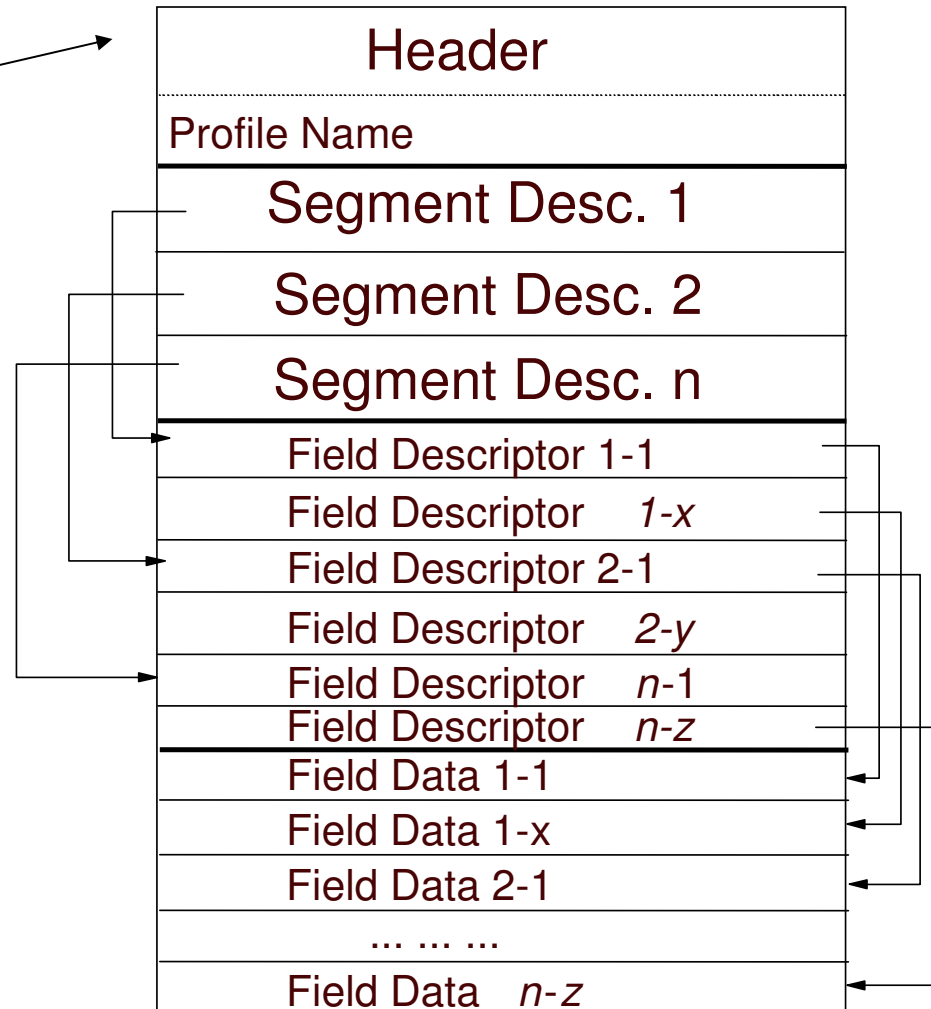
R\_admin extract as a hybrid of a LISTUSER/GRP command and RACROUTE REQUEST=EXTRACT

<b>Like RACROUTE (less filling)</b>	<b>Like a command (tastes great)</b>
Format is architected (i.e. <b><u>supported</u></b> , unlike command output)	Returned data is character (EBCDIC)
Supervisor state caller can bypass authorization	Returned data is 'symmetric'
Runs in caller's address space (much faster than run-command)	Problem state enabled – requires same authorization as command
Can iteratively cycle through profiles	Suppresses fields not displayed by LISTUSER or LISTGRP

---

# Profile extract output format

Parm\_list (input)  
and  
Out\_message\_strings  
(output)



# Input parameter list mapping

0		
8	version	
16		Profile name length
24		
32		Flags 1... Bypass authorization .1...BASE segment only
40		
48		
56		Profile name



# Output parameter list mapping - header

0	"PXTR" eye catcher		Length of output buffer
8	SP	version	... Class
16	... name		Profile name length
24			
32			Flags – cleared!
40	Number of segments		
48			
56			Profile name

---

# Output parameter list mapping – segment descriptor

0	Segment name (e.g. “BASE”, “TSO”, etc) padded	
8	Flags (none currently)	Number of fields
16		Offset to 1 <sup>st</sup> field descriptor
24		
32		

40 Start of next segment descriptor, or, if this is the final segment descriptor, then start of the first field descriptor

---

# Output parameter list mapping – field descriptor

0	Field name, padded	
8	Field type	Flags
16	Length of data	
24	Offset to field data	
32		
40		

Start of next field descriptor, or, if this is the final field descriptor, then start of field data

---

# Repeat Fields

- N-dimensional repeating data fields. E.G.
    - Class authority (CLAUTH) – 1-dimensional
    - Group connection in user profile – 15-dimensional
  - Header field descriptor with unique name identifies
    - Number of occurrences of repeat field
    - Number of elements (dimension) in field
  - Subsequent field descriptors for each constituent field, repeated as necessary
-

# Output parameter list mapping – repeat field header descriptor

0	Field name, padded	
8	Field type	Flags
16	<b>Number of repeat field occurrences</b>	
24	<b>Number of elements (subfields) in repeat field</b>	
32		
40		Start of first subfield descriptor

---

# Repeat Field Schematic Example 1:

## Class authority

CLCNT	1-D	3 occurrences
CLAUTH	8 bytes	Offset to 1 <sup>st</sup> class
CLAUTH	4 bytes	Offset to 2 <sup>nd</sup> class
CLAUTH	8 bytes	Offset to 3 <sup>rd</sup> class

---

Data: FACILITY USER UNIXPRIV

## Repeat Field Schematic Example 2: Group profile member list

CONNECTS	2-D	3 occurrences
GUSERID	5 bytes	Offset to 1 <sup>st</sup> user ID
GAUTH	4 bytes	Offset to 1 <sup>st</sup> authority
GUSERID	5 bytes	Offset to 2 <sup>nd</sup> user ID
GAUTH	3 bytes	Offset to 2 <sup>nd</sup> authority
GUSERID	3 bytes	Offset to 3 <sup>rd</sup> user ID
GAUTH	6 bytes	Offset to 3 <sup>rd</sup> authority

Data: LARRY JOIN CURLY USE MOE CREATE

---

# RACSEQ – Sample TSO command

- Uses R\_admin extract functions to display user, group, or connection attributes
  - Structured output format lends itself to use with REXX OUTTRAP
  - Syntax:
    - RACSEQ CLASS(class) PROFILE(profile)
      - Profile is case-sensitive
  - See RACF web page
-



---

```
RACSEQ CLASS(GROUP) PROFILE(RAPTORS)
Displaying profile RAPTORS in class GROUP. Segments:02
Segment: BASE Fields:08
SUPGROUP:SYS1
CREATDAT:04/18/06
OWNER :IBMUSER
TERMUACC:FALSE
DATA :BIRDS OF PREY KNOW THEY'RE COOL
Repeat field:SUBGRPCT Subfields:01 Occurrences:0004
SUBGROUP:HAWKS
-----
SUBGROUP:FALCONS
-----
SUBGROUP:EAGLES
-----
SUBGROUP:OWLS
-----
Repeat field:CONNECTS Subfields:02 Occurrences:0007
GUSERID :BRUCE
GAUTH :CONNECT
-----
GUSERID :KESTREL
GAUTH :USE
-----
GUSERID :OSPREY
GAUTH :USE
-----
GUSERID :REDTAIL
GAUTH :JOIN
-----
GUSERID :SAWWHET
GAUTH :CREATE
-----
GUSERID :HARRIER
GAUTH :USE
-----
GUSERID :SNOWY
GAUTH :USE
-----
UNIVERSL:FALSE
Segment: OMVS Fields:01
GID :4
```

---

## RACSEQ – Sample output

---

## “Next” requests

- For users and groups (not connections), you can iterate through the profiles by providing a starting value for profile name
    - Next name is returned, similar to ICHEINTY NEXT or RACROUTE REQUEST=EXTRACT TYPE=EXTRACTN
  - The output of the  $n$ th request can be used as the input of the  $n+1$ th request
    - You need only re-specify flags, if desired
-

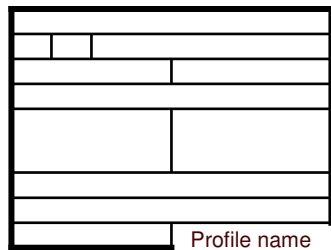
---

## “Next” processing

1. Build the plist header. Specify a profile name of a single blank to start at the top.
  2. Call IRRSEQ00 passing the plist in the Parm\_list parameter. Output returned in Out\_message\_strings parameter.
  3. Free original (or  $n-1$ ) plist.
  4. Process the output as appropriate.
  5. (Re)set header flags, as appropriate
  6. Call IRRSEQ00 with  $n-1$  output as  $n$  input.
  7. Iterate at step 3 until finished (RC 4/4/4).
-

# “Next” Processing (with pictures)

Build Plist header

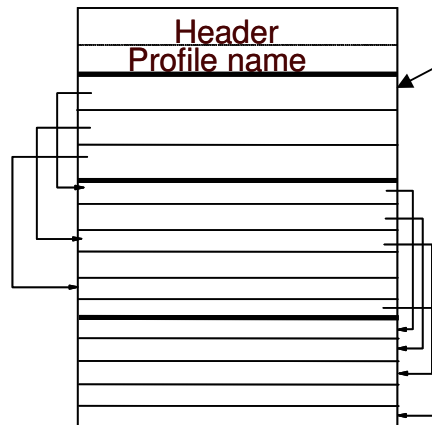


Call R\_admin

```
CALL IRRSEQ00, (Work_area,  
                ALET, SAF_return_code,  
                ALET, RACF_return_code,  
                ALET, RACF_reason_code,  
                Function_code,  
                Parm_list,  
                RACF_userID,  
                ACEE_ptr,  
                Out_message_subpool,  
                Out_message_strings  
                ), VL
```

Free previous storage

Process output



Until done (SAF RC4, RACF RC4, RACF reason code 4 means no more profiles)

---

# SETROPTS Reporting Functions



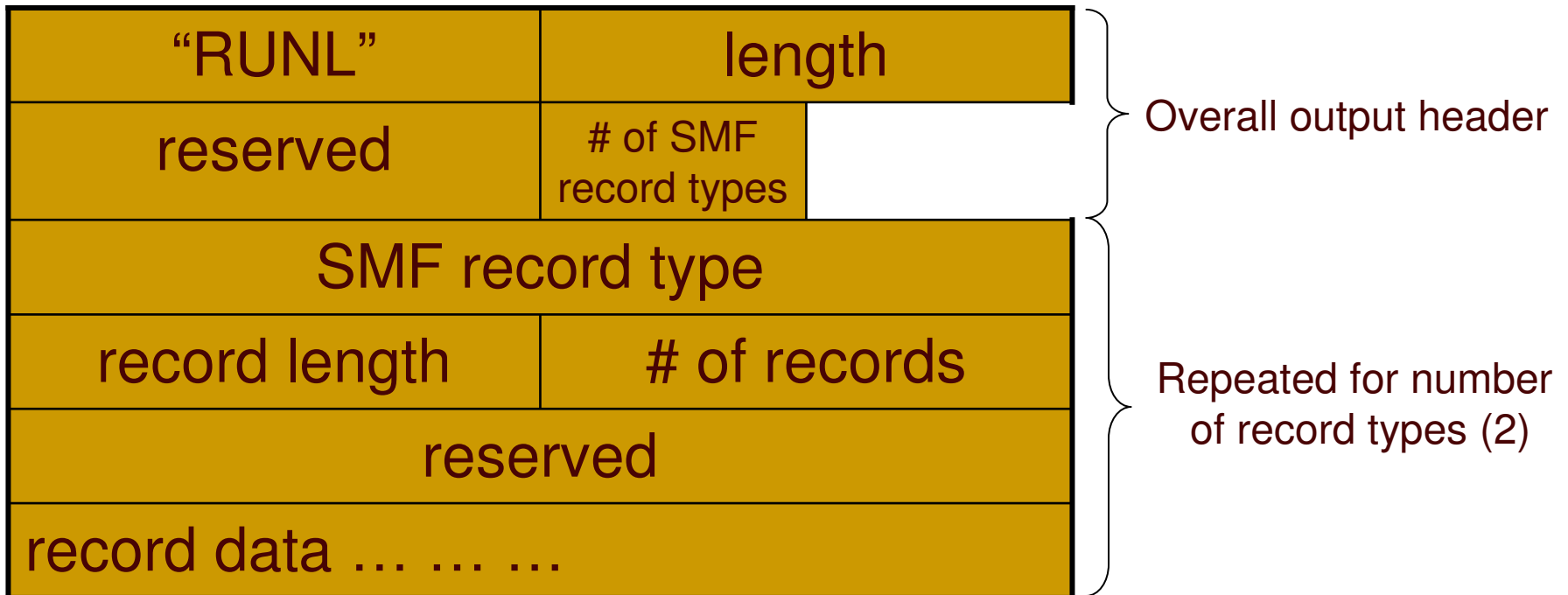
# SETROPTS reporting functions

- Retrieve SETROPTS settings in one of two formats
  - SMF Unload (Type 81)
  - SETROPTS input format (tokenized)
    - Not the same as R7 extract format
      - Sorry!
- Very simple: no input parameter list required

Function codes	Authorization	RACF address space required
22, 23	<ul style="list-style-type: none"><li>■ Supervisor state</li><li>■ SETROPTS LIST authority *not* checked</li></ul>	No

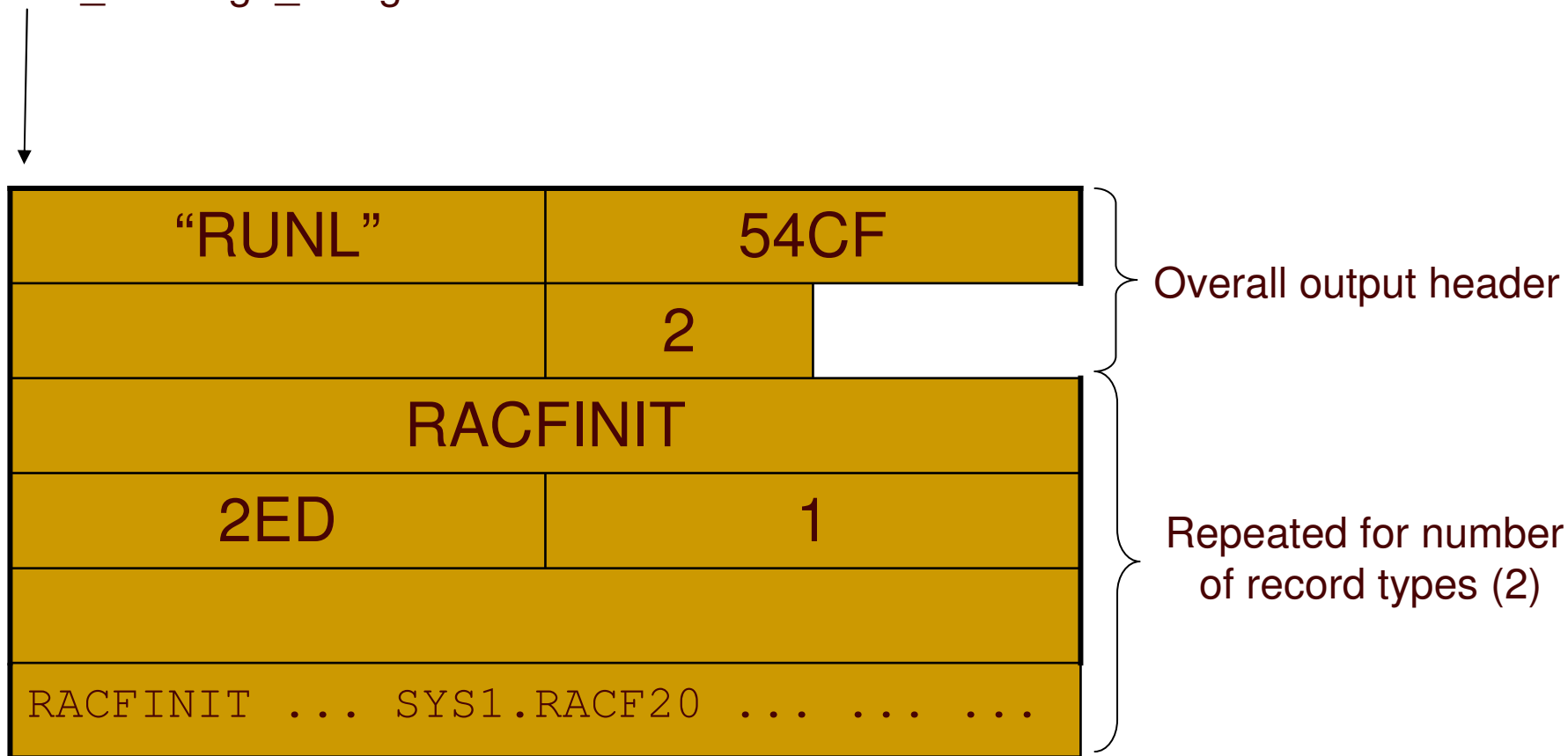
# SETROPTS unload format

Out\_message\_strings



# SETROPTS unload format: Example

Out\_message\_strings





---

# Password Envelope Retrieval



# Password envelope retrieval

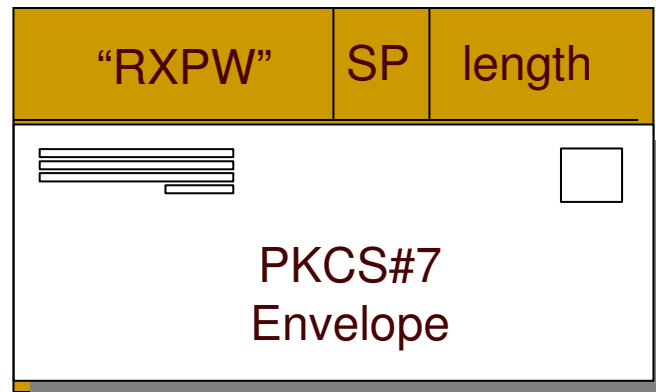
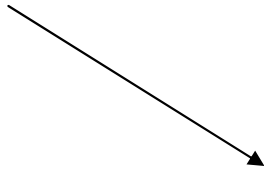
- Rather specialized (and sensitive) little function to extract a specified user's PKCS#7 password envelope
  - From which the clear-text can be recovered
- Intended for password synchronization applications
  - Exploited by Tivoli Directory Integrator

Function code	Authorization	RACF address space required
24	<ul style="list-style-type: none"><li>■ Supervisor state</li><li>■ FACILITY - IRR.RADMIN.EXTRACT.PWENV (READ)</li></ul>	Yes

---

# Password envelope format

Out\_message\_strings



---

# Recap

- API overview
  - Description of each of the functions
    - Run a RACF command
    - Extract user or group profile information
    - Extract SETROPTS settings
    - Retrieve a password envelope
-

---

# References

- RACF Callable Services
  - Command Language Reference
    - [http://publibz.boulder.ibm.com/cgi-bin/bookmgr\\_OS390/Shelves/ICHZBK80](http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/Shelves/ICHZBK80)
  - RACF Downloads page – Sample R\_admin extract program (RACSEQ)
    - <http://www-03.ibm.com/servers/eserver/zseries/zos/racf/goodies.html>
-