

## **Zone Security Rebuild [szonrbld]**

### **Design Overview**

The security features being added to RMS will be maintained in the batch cycle. With each run, the changes made to the data in RMS will be brought under the security features of RMS through the running of 3 batch programs. Szonrbld.pc will handle the maintenance for the zone security data. Zones will have different update/select attributes for a given user for any of a number of different functional areas like 'Pricing' or 'Clearances'. For each run, the program will use the security data defined for the user/group/functional area/pricing zone to define whether a user can select or update every zone covered by the defined rules.

It is possible that users can be on the system at anytime, indicated by the Batch with Online Users indicator on the system\_options table. Security table views were created such that while the program is rebuilding the security, the on-line application will point to the permanent table with the old security. If the Batch with Online Users indicator is set to 'Y', once the new security has been built a batch program (prepost.pc – *szonrbld post*) will switch the on-line application view to the permanent table with the new security.

The functional document describes the architecture of the security features and how it works. If there are conflicting security definitions for a zone because a user is associated with more than one group, the user is, as expected, granted the capability. It will be multi-threaded by zone, and use restart\_recovery.

TABLE	INDEX	SELECT	INSERT	UPDATE	DELETE
SEC_USER_GROUP	No	Yes	No	No	No
SEC_GROUP_ZONE_MATRIX	No	Yes	No	No	No
V_RESTART_ZONE_GROUP_ID	No	Yes	No	No	No
SYSTEM_VARIABLES	No	Yes	No	No	No
SEC_USER_ZONE_MATRIX_A	No	No	Yes	No	No
SEC_USER_ZONE_MATRIX_B	No	No	Yes	No	No
SYSTEM_OPTIONS	No	Yes	No	No	No

### **Scheduling Constraints**

Processing Cycle: Daily

Scheduling Diagram: Must run batch program prepost.pc with parameters *szonrbld pre* , *szonrbld.pc* and *prepost.pc* with parameters *szonrbld post* in series.

Pre-Processing: Run prepost with parameters: *szonrbld pre*

Post-Processing: Run prepost with parameters: *szonrbld post*

Threading Scheme: Zone\_group\_id

### **Restart Recovery**

The logical unit of work for zone security rebuild will be the user-functional area (column\_code)-zone\_group\_id-zone\_id. Restart/recovery will be based on the user-functional area (column\_code)-zone\_group\_id-zone\_id. The commit\_max\_ctr field should be set to prevent excessive rollback space usage.

### **Program Flow**

N/A

### **Shared Modules**

N/A

### **Driving Cursor:**

```
SELECT u.user_id,
       z.column_code,
       z.zone_group_id,
       z.zone_id,
       MAX(z.select_ind),
       MAX(z.update_ind)
FROM sec_user_group u,
     sec_group_zone_matrix z,
     v_restart_zone_group_id v
WHERE u.group_id = z.group_id
AND v.driver_value = z.zone_group_id
AND v.num_threads = :pi_restart_num_threads
AND v.thread_val = :pi_restart_thread_val
AND (u.user_id > NVL(:ps_restart_user, '-999')
     OR (u.user_id = :ps_restart_user
         AND (z.column_code > :ps_restart_column_code
              OR (z.column_code = :ps_restart_column_code
                  AND (z.zone_group_id > :ps_restart_zone_group_id
                      OR (z.zone_group_id = :ps_restart_zone_group_id
                          AND (z.zone_id > :ps_restart_zone_id)))))))
GROUP BY u.user_id, z.column_code, z.zone_id, z.zone_group_id
```

### **Function Level Description**

Main

Init()

- Check SYSTEM\_VARIABLES.update\_loc\_sec\_ind. If the indicator is not set then the program exit normally without further processing.
- Call retek\_init() to get restart-recover variables.
- Check SYSTEM\_OPTIONS.btch\_w\_usr\_ind to see if the Batch with User Online indicator is 'Y'.

Process()

- Call size\_rule\_array() to allocate memory for arrays that store security rules.
- Open the driving cursor in a while loop. Fetch the data into rule array.
- Call table\_view\_check() to get the permanent table that is set as the current table view.

- Insert records in rule array into the permanent table
- Call restart\_commit.
- End of while loop.

#### Size\_rule\_array()

This function allocates memory for arrays that store security rules based on the maximum commit count set in table restart\_control table. The rule array includes user\_id, column\_code, zone\_group\_id, zone\_id, select\_ind and update\_ind.

#### Insert\_array\_to\_table()

- This function uses array insert to insert the contents in the rule array to the permanent table.
- If SYSTEM\_OPTIONS.btch\_w\_usr\_ind is 'Y' then insert records into the permanent table that is not set as the current table view. Otherwise, insert records into the permanent table that is set as the current table view.

#### Table\_view\_check():

This function returns the permanent table that is set as the current table view.

#### final():

restart/recovery close

### **I/O Specification**

N/A

### **Technical Issues**

N/A