Bypass saving records in VFA for find processing (APAR PJ48052)

System Control Program

Gabriel Nieves

2025 TPF Users Group Conference
May 05-07, Austin, Texas

IBM Z

IBM

Problem Statement

Read-heavy utilities that reference VFA candidate records can increase response time for transactional workloads because VFA performance is impacted.

Users

Sophie is responsible for developing a long-running utility that inspects a large database with VFA candidate records.

Sophie
System Programmer

Derrick is responsible for scheduling system utility runs and monitoring the system performance.

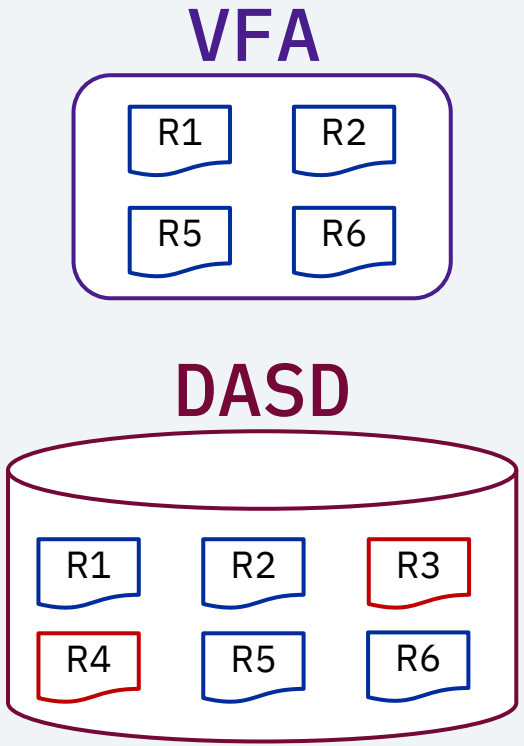
Derrick
Operator

As-Is User Story

- The long-running utility reads numerous VFA candidate records in one or more large database as part of its processing.
- Sophie is worried about the impact on transactional workload while running the utility.
- Derrick must be careful that the utility run is not scheduled during peak hours when VFA performance is most critical.

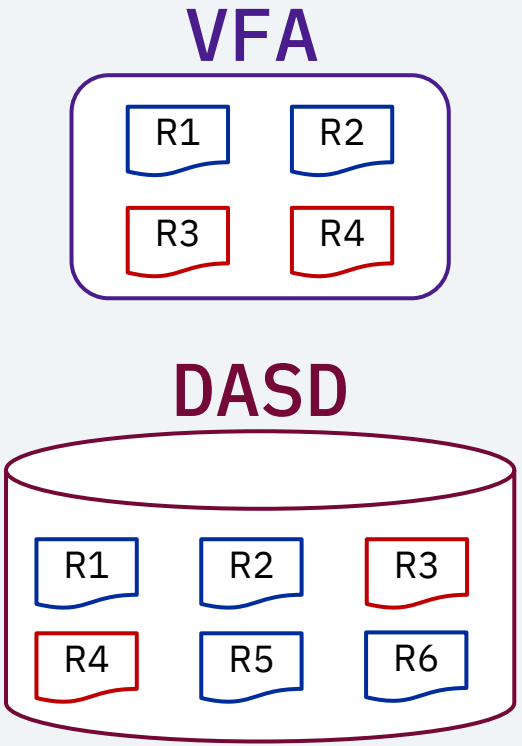
As-Is User Story (Cont.)

- The blue records (R1, R2, R5, R6) are frequently accessed by transactional workload.
- The red records (R3, R4) are rarely accessed during normal activity.
- The blue records should always stay in VFA to have the best performance for transactional workload.



As-Is User Story (Cont.)

- When running the long-running utility, some blue records are pushed out of VFA.
- Subsequent transactional workload that requires those records would need to retrieve them from DASD.
- Both workloads are competing for limited VFA resources.



Pain Points

- If the utility reads records that are not actively being referenced by the transactional workload, VFA performance can be impacted when other records are pushed out of VFA.
- APAR PJ46019 added the VFAFILL parameter to the FINDC macro, but it does not support other FIND macros and requires code changes wherever the macro is used.

Value Statement

A read-heavy utility can run during peak hours, with no code changes, while not increasing response time of transactional workloads.

Technical Details

- With PJ46019, FINDC VFAFILL=NO causes the record to not be copied to VFA if not already in VFA.
- With PJ48052, new ECB attribute VFAFILL provides the same behavior when you use any general FIND APIs.
- By default, an ECB has the VFAFILL attribute set to YES and VFA candidate records are copied to VFA.
- EASETC APIs can be used to change the attribute setting.
  - Example: **`EASETC VFAFILL=NO,INHERIT=YES`**

Technical Details (Cont.)

- With the –VFN console command prefix, VFAFILL attribute can be set to NO for the created ECB with no code changes.
- Child ECBs will also inherit this ECB attribute setting.
- Multiple command prefixes can be used at the same time.
- The following example starts the chain chase utility as a low priority utility with the VFAFILL attribute set to NO:

```
-VFN/-LP/ZCHCH F4034035 F 020 HDR-4
```

Technical Details (Cont.)

- ZCFIL COPY, ZCFIL SAVE, and ZIFIL FILE commands were updated to set the VFAFILL attribute to NO.
- EASETC macro and new tpf_easetc_ext function can now save and restore the ECB attribute settings in a storage location specified by the application.

Technical Details (Cont.)

The following C example would save the previous ECB attribute settings before turning on the low priority attribute:

```
tpf_easetc_parms easetc_parms;
unsigned long prev_ecb_attributes;
memset(&easetc_parms, 0, sizeof(tpf_easetc_parms));
easetc_parms.attribute = TPF_EASETC_LOWPRIORITY;
easetc_parms.options = TPF_EASETC_SET_ON+TPF_EASETC_INHERIT_YES;
easetc_parms.save_prev = &prev_ecb_attributes;
tpf_easetc_ext(&easetc_parms);
```

Technical Details (Cont.)

The following C example would restore the previously saved ECB attribute settings from the previous example:

```
…
memset(&easetc_parms, 0, sizeof(tpf_easetc_parms));
easetc_parms.use_prev = &prev_ecb_attributes;
tpf_easetc_ext(&easetc_parms);
```

Conclusion

APAR PJ48052 (January 2025) delivers support for reducing the amount of VFA resources used by utilities that read a large number of records from DASD.

Thank you