z/TPFDF Status Update
Database Subcommittee

Chris Filachek
z/TPF and z/TPFDF Architecture & Development

IBM **z/TPF**
April 4, 2017

# z/TPFDF Delivered Enhancements

Include Referenced Subfiles on Delete

A TPF programmer can use the delete APIs with the INCLUDE parameter to release referenced subfiles without requiring extra application logic.

- Delete operations ignored the INCLUDE parameter for files using algorithm #TPFDB0D
    - Referenced subfiles were not deleted
    - Caused lost pool records

- Changed so the INCLUDE parameter is honored on DBDEL / dfdel APIs for algorithm #TPFDB0D

- APAR PI59102 on PUT 13

A database administrator can encrypt data-at-rest in z/TPFDF files and protect sensitive customer information without requiring any application changes.

- Encrypted on DASD, in VFA, and in z/TPFDF Cache

- Protected using AES encryption algorithms in Cipher Block Chaining (CBC) mode
    - AES-128 CBC
    - AES-256 CBC

- Optional: Identify data corruption using data integrity verification
    - Uses SHA-256 message digests
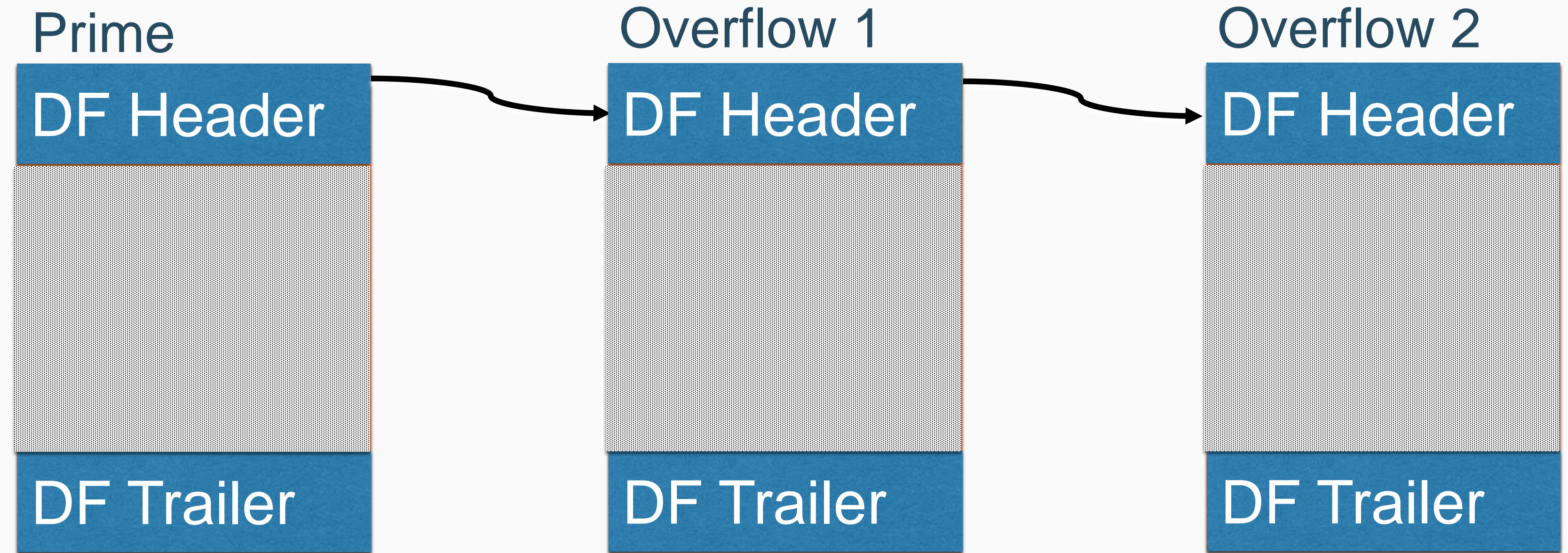
- APAR PI56476 on PUT 13
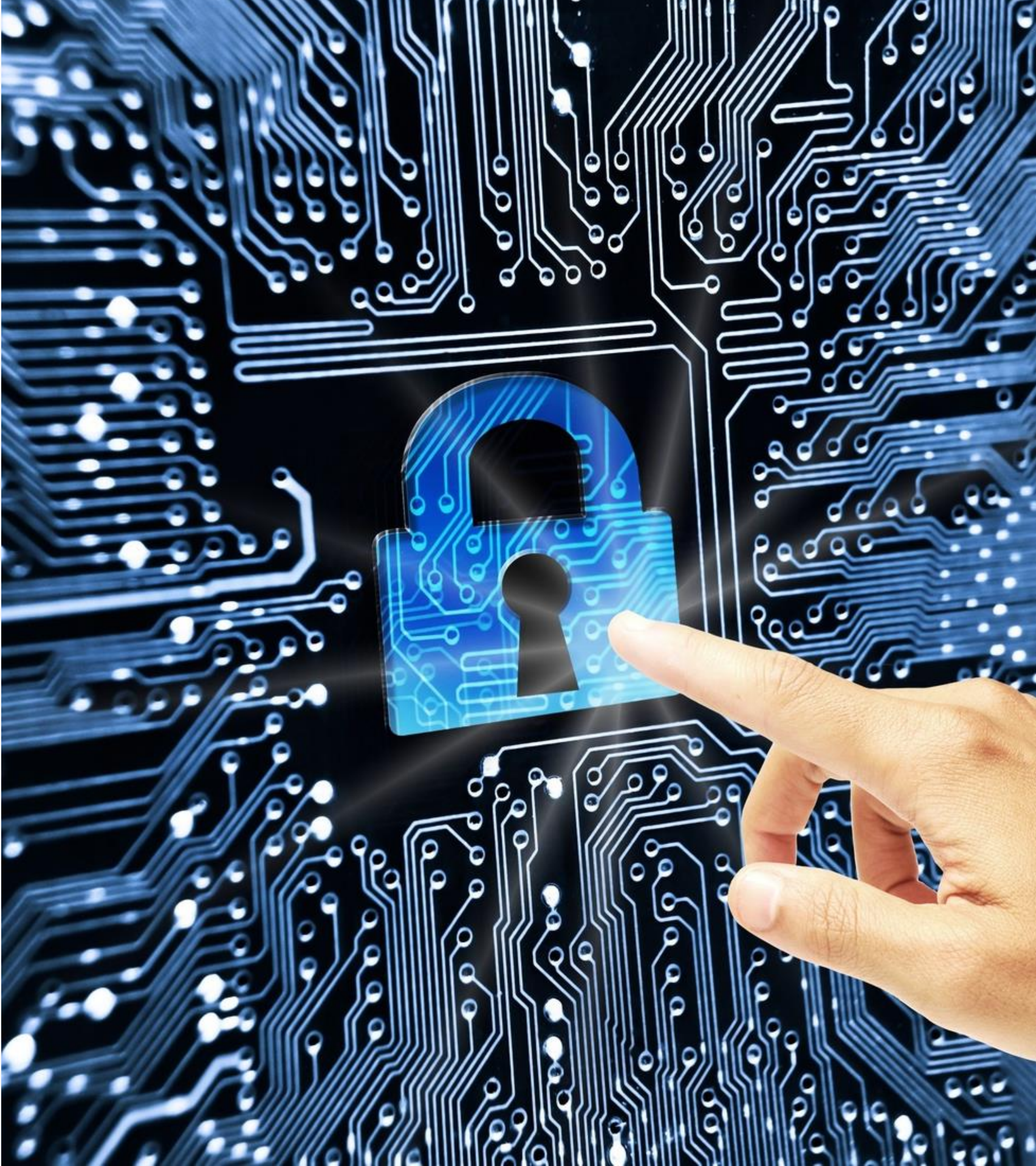
z/TPFDF Encryption – What is Encrypted?

Encrypted

- Standard data areas (LRECs)
- First block of LLR (MLL)

Not Encrypted

- z/TPFDF standard headers & trailers
- LLR past MLL
- B+Tree nodes

Prime

DF Header
DF Trailer

Overflow 1

DF Header
DF Trailer

Overflow 2

DF Header
DF Trailer

No application changes required!

- Data is automatically encrypted/decrypted by z/TPFDF APIs, commands, and utilities
- Encryption settings managed through commands

Migrate Existing Subfiles

1. Update DBDEF to allow encryption

2. Migrate to new trailer format
   - CRUISE with PACK parameter

3. Define keys in the z/TPF Symmetric Keystore
   - ZKEYS command

4. Define options and enable encryption
   - ZUDFM ENCRYPT command
   - New and updated subfiles encrypted

5. Encrypt all subfiles
   - CRUISE with PACK parameter

No downtime required!

Manage Encryption

Encryption keys and options

- Change key values in the z/TPF Symmetric Keystore using ZKEYS command

- Manage key names and options for each z/TPFDF file using ZUDFM ENCRYPT command

- Propagate changes across all subfiles using CRUISE with PACK parameter

Transformation Engine (TE) eligible

- Encryption, decryption, and data verification processing
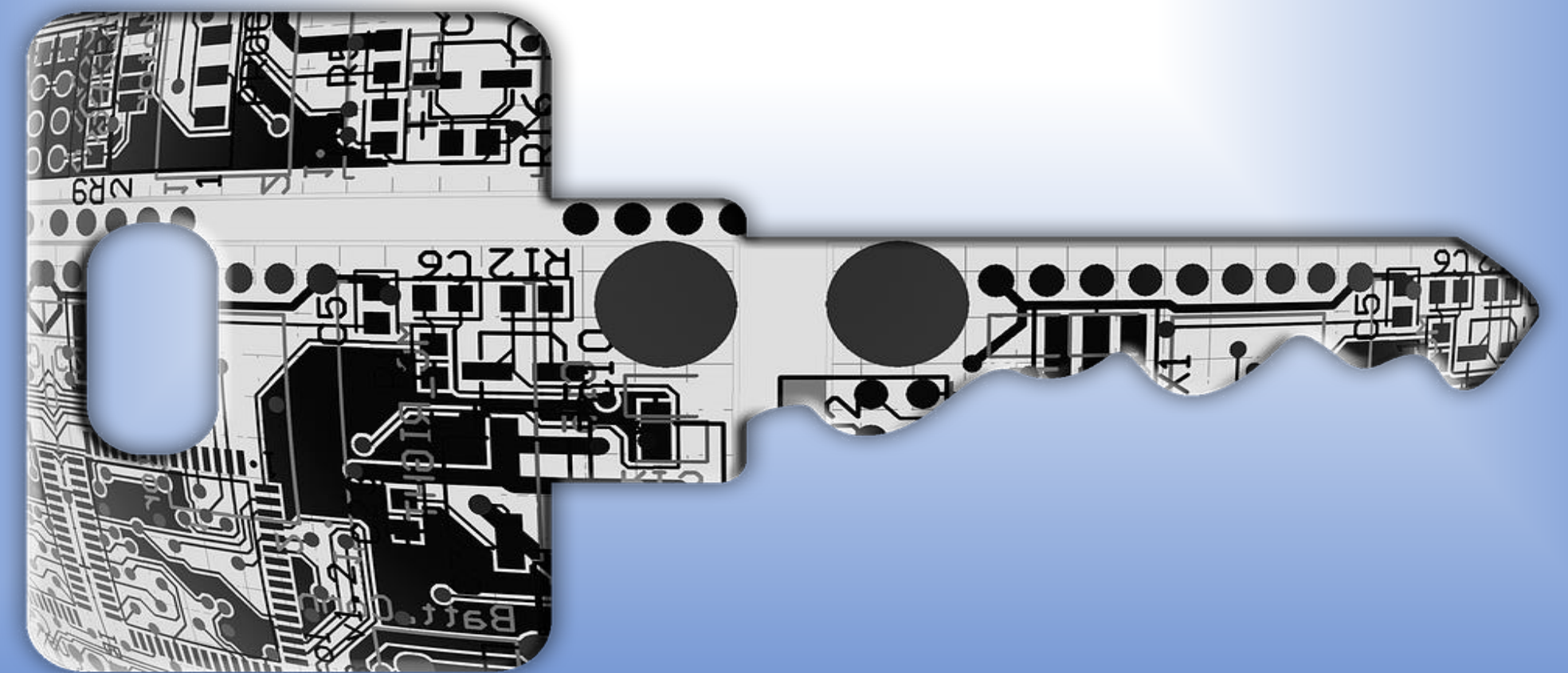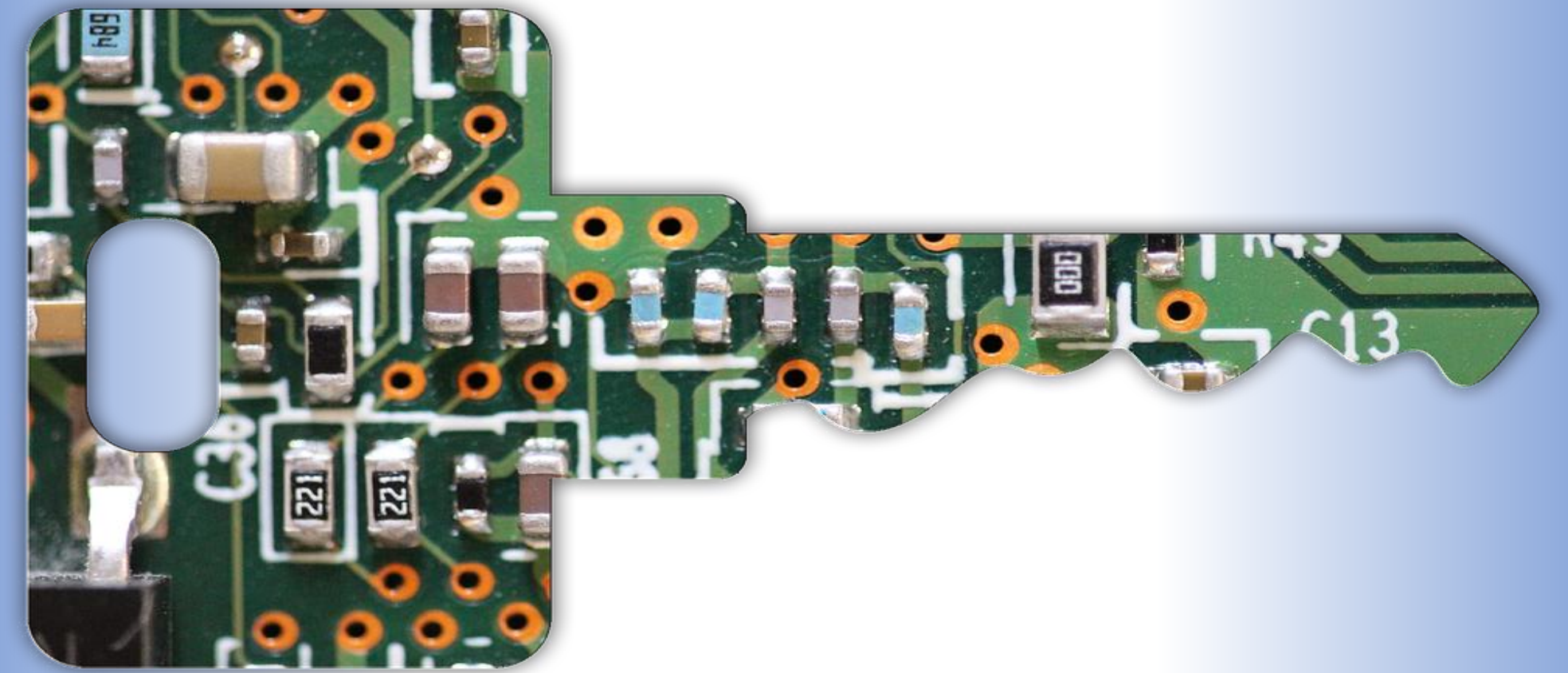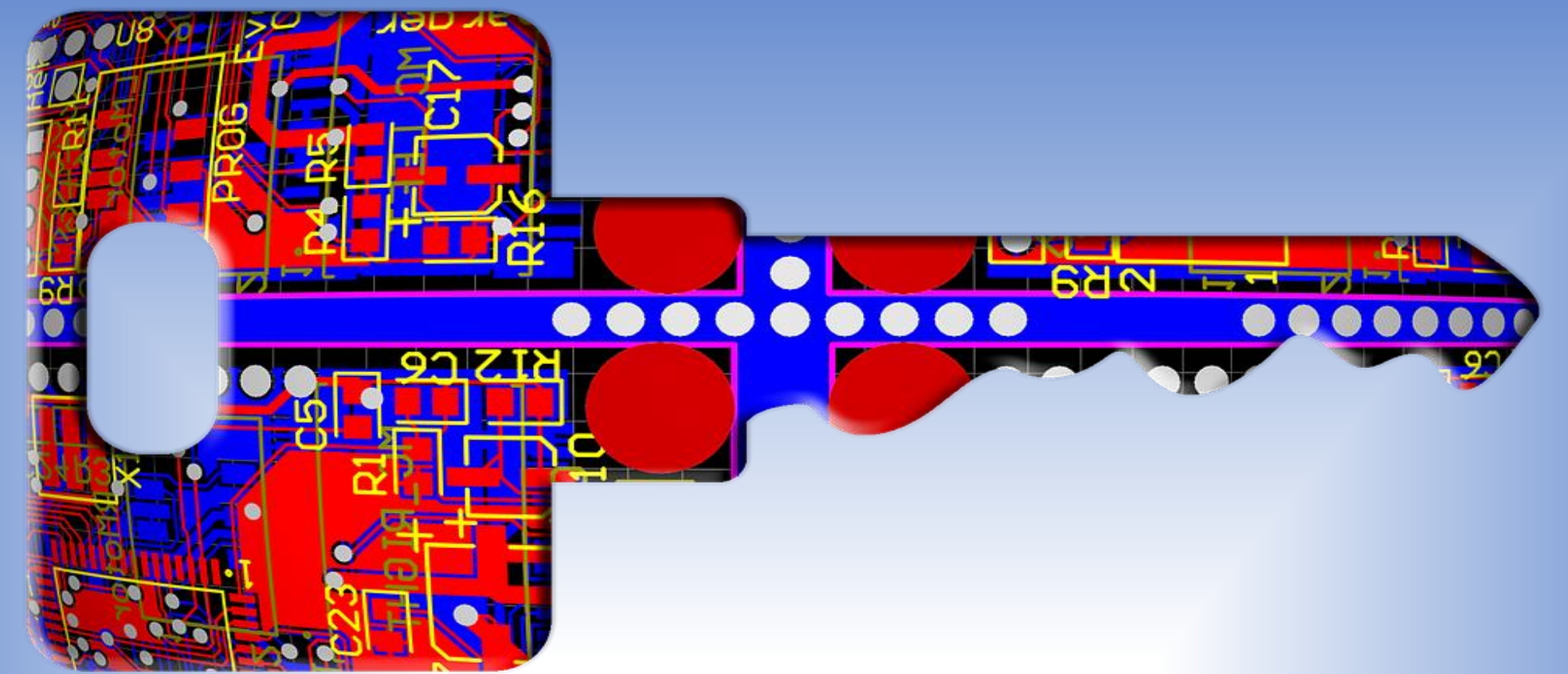
z/TPFDF **Future** Enhancements

Disclaimer

Any reference to future plans are for planning purposes only.

IBM reserves the right to change those plans at its discretion.

Any reliance on such a disclosure is solely at your own risk.

IBM makes no commitment to provide additional information in the future.

File address limitations

- FARF5 has a limit of 4 billion addresses

- Shared between pool and fixed

Large fixed record databases

- Can use 100's of millions of addresses in a single record type

- Top-level indexes

- Hash-table type databases

FARF6 Fixed for z/TPFDF

- Support databases using FARF6 Fixed Records
  - Enable FARF6 addressing for z/TPFDF - ZMODE T command
  - z/TPFDF FARF6 headers and trailers are required for FARF6 fixed databases

- New FARF6 fixed record databases

- No migration planned for existing fixed record databases
  - What are the migration use cases?
  - Join as a sponsor user!
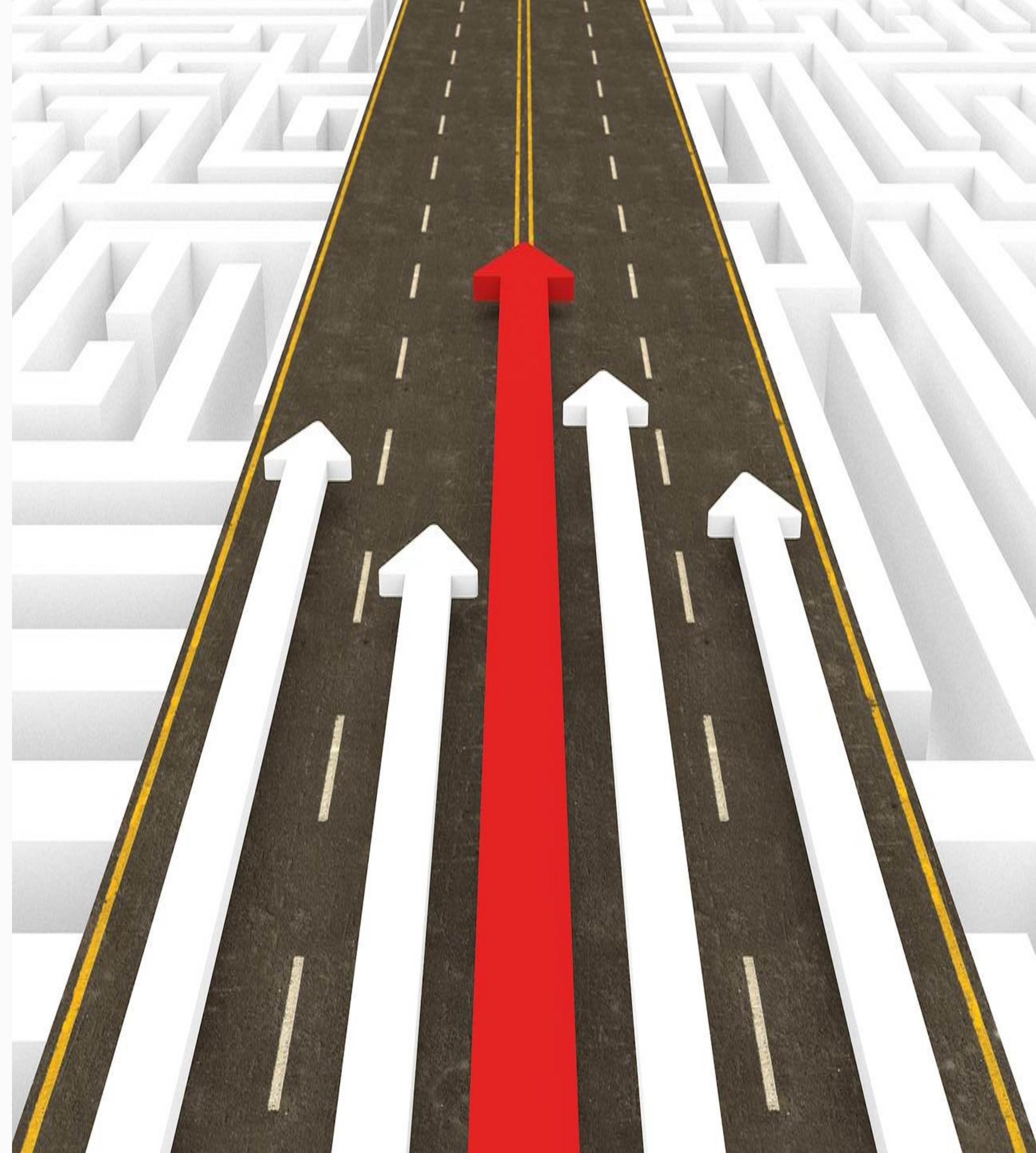
Algorithm Caching – As Is

- Algorithm strings are used to traverse indexes and find the detail subfile
  - Example: Last name plus phone number used to find my PNR
- Each level of an index must be read and processed before getting to the next level and the detail subfile
  - May require multiple I/O's to get to the detail subfile
  - VFA and z/TPFDF cache reduce physical I/O
  - Still requires processing each index level

A database administrator can improve the performance of z/TPFDF index lookups by caching indexing results and without making any application changes.

- First read using algorithm string
  - Read indexes (existing path)
  - Cache the algorithm string (Last name plus phone number) and the prime file address of the PNR
- Subsequent reads query algorithm string in cache and use cached prime file address

Algorithm Caching – Current Thinking

➢ Benefit increases as the number of index levels increase

➢ Assumptions

- The same detail subfile is read multiple times using the algorithm string
  - Populate cache on first read
  - Use cache on subsequent reads
  - No benefit if the algorithm string is only used once
- Cache is in local memory and is processor unique
  - A processor cannot use items in cache on other processors

➢ Requirements

- Loosely coupled systems would require coupling facility (CF)
  - Invalidate cached results across complex as indexes are updated or deleted

➢ Interested in being a sponsor user?

- Do you have databases and indexes that may benefit?
- Looking for sponsor users!

- LREC ID's are chosen when a z/TPFDF file is first designed
  - Most frequently used LRECs are organized towards the front of the subfile
- Databases evolve over time...
  - New data is added to existing LRECs
  - Additional LREC IDs are added
  - Use patterns of LRECs change
- No clear way to validate existing LREC organization within a z/TPFDF file

A database administrator can use tools to review z/TPFDF file LREC ID access patterns without impacting the existing workload or requiring application changes.

- Sample accesses by LREC ID for one or more z/TPFDF files

- Offline report displays access statistics for each LREC ID

- Based on report, database administrator may be able to recommend changes to application groups
  - How would information be used?
  - Would applications group make changes?
  - Interested in being a sponsor user?

THANK YOU

Questions or comments?

Chris Filachek
z/TPF and z/TPFDF Architecture & Development

IBM **z/TPF**
April 4, 2017

trademarks

IBM, the IBM logo, ibm.com and Rational are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

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.

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.

This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.