z/TPF EE V1.1
z/TPFDF V1.1
TPF Toolkit for WebSphere® Studio V3
TPF Operations Server V1.2

IBM Software Group

TPF Users Group Spring 2007

z/TPF secure key management

Subcommittee presentation, part 2

AIM Enterprise Platform Software
IBM z/Transaction Processing Facility Enterprise Edition 1.1.0

Operator procedures for backing up and restoring the keystore

Backing up keystore data

- ZKEYS BACKUP command
- Copies the master keystore information to the specified file in the z/TPF file system
  - Information in the master keystore is encrypted; therefore, information in the file is also encrypted
  - Keystore information is validated as well
- You can FTP a copy of the backup file to another system for disaster recovery purposes
- Optional PASSWORD parameter on ZKEYS BACKUP command allows you to control access to the backup file

ZKEYS BACKUP PATH-/keys/keysback.fil PASSWORD-MYSECRET

KEYS0004I 09:41:01 MASTER KEYSTORE BACKUP STARTED
KEYS0005I 09:41:02 MASTER KEYSTORE BACKUP COMPLETED

Validating backup copy of keystore data

- ZKEYS VALIDATE command
- Used to verify that the specified keystore backup file is still usable and current
  - Makes sure the backup file has not been corrupted
  - Checks to see if any changes have been made to the master keystore since the backup file was created
- PASSWORD parameter is optional
  - If specified, it must be the same password that was entered on the ZKEYS BACKUP command that created the backup file

ZKEYS VALIDATE PATH-/keys/keysback.fil

KEYS0008I 15:41:01 ZKEYS VALIDATE PROCESSING STARTED
KEYS0009I 15:41:02 ZKEYS VALIDATE PROCESSING COMPLETED

Restoring the master keystore

- ZKEYS RESTORE command
- Rebuilds the master keystore using the information in the specified keystore backup file
- If a password was specified on ZKEYS BACKUP command that created the backup file, you must specify the same password on the ZKEYS RESTORE command
- This command does ***not*** affect the contents of the memory keystore on any processor

ZKEYS RESTORE PATH-/keys/keysback.fil PASSWORD-MYSECRET

KEYS0010I 17:32:02 ZKEYS RESTORE PROCESSING STARTED
KEYS0011I 17:32:03 ZKEYS RESTORE PROCESSING COMPLETED

Keystore corruption

Restart and memory keystore corruption

- During restart after an IPL, the master keystore is read from DASD and used to build the memory keystore
  - The master keystore information is validated as part of this process
  - If the master keystore is found to be corrupted:
    - You will need to restore the master keystore (using ZKEYS RESTORE command)
    - The system will continue restart processing and be allowed to reach 1052 state so that you can FTP a keystore backup file into z/TPF if necessary
- Each entry in the memory keystore is validated before it is used
  - If the memory keystore is found to be corrupted
    - Catastrophic system error is taken causing the memory keystore to be rebuilt from the master keystore after the IPL

Master keystore corruption after system restart

- If secure key restart processing completed successfully, that means the memory keystore is usable
- The master keystore is validated:
  - During the processing of most ZKEYS commands
  - Periodically by the Keystore Monitor
    - How frequently the monitor runs is defined by the KEYSVAL parameter on the SKEYS macro in SIP
    - Use the ZKEYS KEYSVAL command to dynamically change how frequently the keystore monitor runs
- If the master keystore is found to be corrupted
  - The master keystore is rebuilt using the memory keystore information
  - Applications can continue to encrypt/decrypt data while the master keystore is being rebuilt

Keystore backup and restore considerations

- Keystore backup recommended procedures:
  - Backup the master keystore (ZKEYS BACKUP) whenever keys are added, activated, or deactivated
    - You are required to do a backup before activating new keys
  - Keep at least one current backup copy in the z/TPF file system
  - Keep a backup copy on at least one remote system as well
    - Use FTP to send a copy of the backup file to the remote system
- Options for recovering a corrupted master keystore
  - The system will automatically rebuild the master keystore using the memory keystore if the memory keystore is built (secure key restart completed successfully) and the memory keystore is not corrupted
  - Issue ZKEYS RESTORE to restore the master keystore using a backup copy in the z/TPF file system
  - FTP a backup copy saved on a remote system to the z/TPF file system, then issue ZKEYS RESTORE

Displaying keystore information

ZKEYS DISPLAY commands

- ZKEYS DISPLAY SUMMARY
  - Shows information about keystore resources
- ZKEYS DISPLAY STATS
  - Shows statistical information about secure key operations at a system wide level
- Other ZKEYS DISPLAY commands options
  - Displays information about one or more keys in the keystore
    - For example, display all keys, a specific key, active keys, all keys with the same encryption key name
  - Display includes statistical information about key usage on this processor

ZKEYS DISPLAY SUMMARY example

```
ZKEYS DISPLAY SUMMARY
KEYS0024I 17.34.09 ZKEYS DISPLAY SUMMARY

KEYSTORE ENTRIES IN USE : 39
MAXIMUM ENTRIES IN MASTER KEYSTORE : 975
HIGHEST MASTER KEYSTORE RECORD ORDINAL IN USE : 2
MAXIMUM ENTRIES IN MEMORY KEYSTORE : 100
MAXIMUM ENTRIES IN MEMORY KEYSTORE ON NEXT IPL : 100
LAST UPDATE TO MASTER KEYSTORE : 17.16.37 - MAR 19, 2007
LAST UPDATE IN KEYSTORE BACKUP : 17.16.37 - MAR 19, 2007
MASTER KEYSTORE VALIDATION INTERVAL : 1

END OF DISPLAY
```

ZKEYS DISPLAY SUMMARY example description

```
KEYSTORE ENTRIES IN USE : 39
MAXIMUM ENTRIES IN MASTER KEYSTORE : 975
MAXIMUM ENTRIES IN MEMORY KEYSTORE : 100
MAXIMUM ENTRIES IN MEMORY KEYSTORE ON NEXT IPL : 100
```

39 = number of keys defined in the keystore
975 = maximum number of keys that will fit in the master keystore based on the number of #IKEYS fixed file records defined
100 = value of KEYSENT in CTKC when this processor IPL'd
100 = current value of KEYSENT in CTKC

ZKEYS DISPLAY SUMMARY example description

```
LAST UPDATE TO MASTER KEYSTORE : 17.16.37 - MAR 19, 2007
LAST UPDATE IN KEYSTORE BACKUP : 17.16.37 - MAR 19, 2007
```

- If the two timestamps above are equal
  - The backup keystore file is current, meaning no changes have been made to master keystore since the last keystore backup (ZKEYS BACKUP) was done

```
MASTER KEYSTORE VALIDATION INTERVAL : 1
```

1 = value of KEYSVAL in CTKC

ZKEYS DISPLAY keys example

```
ZKEYS DISPLAY ENC-AP1EKEY
KEYS0022I 17.17.11 ZKEYS DISPLAY PROCESSING STARTED

ENC NAME DEC NAME ACT  ACT DATE  CIPHER   ENC/SEC DEC/SEC MAX ENC MAX DEC
-------- -------- --- --------- --------- ------- ------- ------- -------
AP1EKEY  AP1DKY06  Y  13APR2007 TDES        25781   19265   40567   28213
AP1EKEY  AP1DKY05  N  13MAR2007 TDES            0       2       0      18
AP1EKEY  AP1DKY04  N  13FEB2007 TDES            0       0       0       2
AP1EKEY  AP1DKY03  N  13JAN2007 DES             0       0       0       0
AP1EKEY  AP1DKY02  N  13DEC2006 DES             0       0       0       0
AP1EKEY  AP1DKY01  N  13NOV2006 DES             0       0       0       0
END OF DISPLAY
```

Installation procedures

Initialize master keystore

- ZKEYS INITIALIZE command
- Initializes the master keystore
- You need to do this once when you first install the code and should be the only time you use this command in a production system
  - If the master keystore becomes corrupted, restore it (using the ZKEYS RESTORE command) rather than reinitializing the keystore
- Operator will be prompted to confirm the initialization of the master keystore

Installation procedures

1. Install the APAR PJ31450 code
2. Determine how many secure keys you need
3. Define master and memory keystore resources
4. Initialize master keystore
5. Create secure keys and add your existing keys to the keystore
6. Make backup copy of the keystore
7. Activate the keys
8. Make another backup copy of the keystore
9. Assign keys to applications
10. Update secure key authorization user exit
11. Update applications to use secure key APIs

Installation requirements

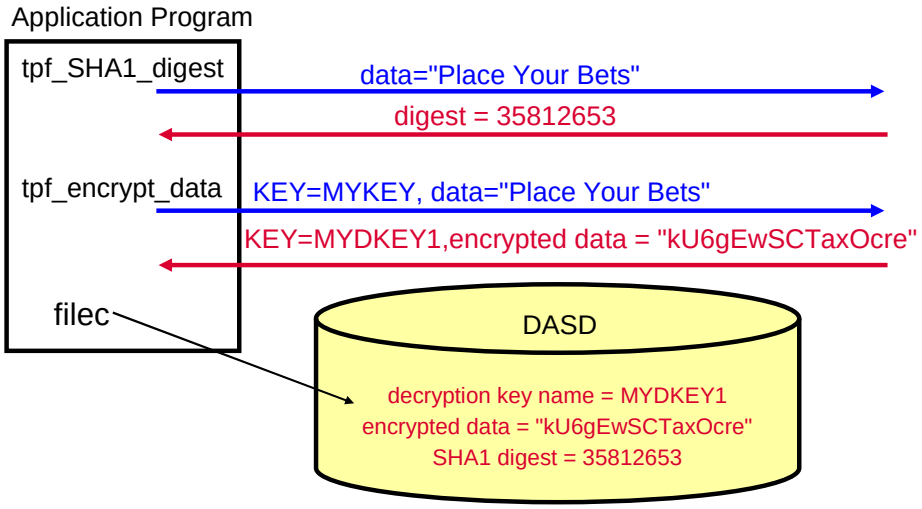
- Enabling secure key support:
  - APAR PJ31450 must be installed on all processors
  - All processors must be z990 or higher
    - Secure key support requires CPACF
  - DES/TDES feature on CPACF must be enabled
- To use AES128 or AES128-CBC ciphers
  - Must be running z9 or higher

Data integrity

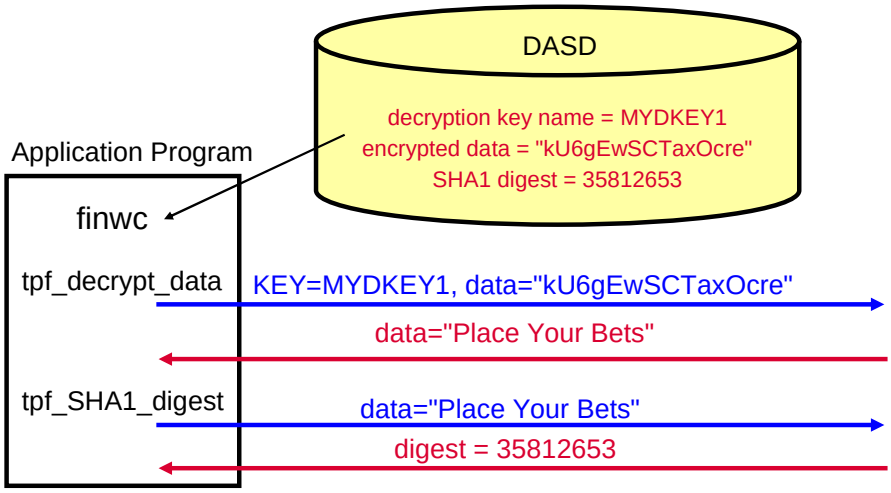
Data integrity

- Recommended usage:
  1. Create a digest of the (unencrypted) data
  2. Save the digest along with the encrypted data
     - Also save what digest algorithm was used in case you change algorithms in the future
  3. Recalculate the digest after data is decrypted and compare to the original digest to verify that the data has not been altered
- Use SHA-1 APIs to calculate digests
  - Provided by z/TPF APAR PJ31336
- Statement of direction to provide SHA-256 APIs

Data encryption example with data integrity



Data decryption example with data integrity

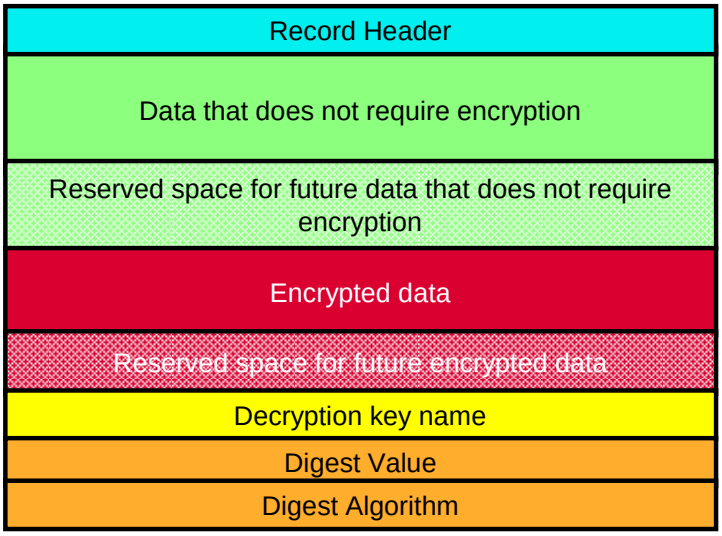


Data record design considerations

Data record design considerations

- For records containing data encrypted at the application level that are stored on DASD, stored on tape, or sent across the network
- Try to group together all data that needs to be encrypted
  - If data to be encrypted is contiguous
    - One tpf_encrypt_data API can encrypt all that data
  - If data to be encrypted is not contiguous
    - Multiple *tpf_encrypt_data* APIs are needed
    - Make sure the application issues all the *tpf_encrypt_data* APIs for a given record without giving up control to prevent the encryption key value from changing in the middle of processing a record
- Reserve space for both regular data (that does not require encryption) and encrypted data
- Reserve 32 bytes for digest to accommodate SHA-256
  - 20 bytes needed for SHA-1 digest

Suggested record layout



Performance data

Secure key performance implications

- Number of keys defined does not impact performance
  - For example, the system can process the same number of secure key APIs per second with 2 keys defined in the keystore as when 2,000 keys are defined
- Number of secure key APIs that can be processed per second grows with the number of I-streams
  - Each I-stream has its own CPACF
  - Roughly linear scaling
- Secure Key versus Clear Key
  - Ran tests with different data sizes first using clear key APIs (*tpf_cryptc*), then with secure key APIs (*tpf_encrypt_data* and *tpf_decrypt_data*)
    - The number of crypto operations (APIs) per second is 2-10% less using secure key compared to clear key

Secure key data operations/second - single I-stream, different ciphers

Data Size	DES	TDES	TDES-CBC	AES128
16	449,292	440,516	434,490	430,800
32	442,148	428,586		422,658
64	433,236	400,434		410,058
256	386,958	310,818	304,200	322,955
1024	274,970	158,508		188,974
4096	124,352	53,780	53,246	69,222
32,768	20,622	7,480		9,785
65,536	10,461	3,753	3,734	4,968
1,048,576	669	236		313

Tests performed on z9 processor - your results may vary

Secure key data operations/second - TDES, multiple I-streams

Data Size	1 I-Stream	2 I-streams	Scaling Factor	5 I-Streams	Scaling	9 I-streams	Scaling
64	400,434	740,449	1.85	1,847,119	4.61	3,339,130	8.34
4096	53,780	105,305	1.96	265,512	4.94	480,407	8.93
65,536	3,753	7,466	1.99	18,762	4.99	33,770	8.99
1,048,576	236	469	1.99	1,176	4.99	2,122	8.99

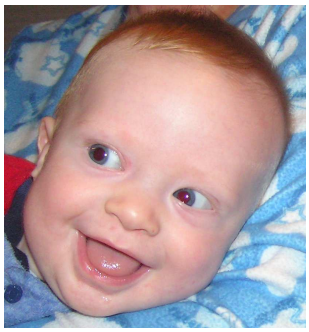
Tests performed on z9 processor - your results may vary

Summary

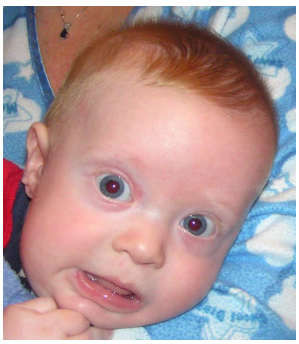
z/TPF secure key management support highlights

- Ability to change encryption key values (and in some cases upgrade the cipher) without requiring any application program changes
- Can scale to hundreds of thousands of crypto operations per second
- Ability to control and log key usage by applications
- Ability to control who can create keys (operators and applications)
- Ability to backup and restore keys
- Ability to migrate your existing keys to this support
- Safeguards to prevent a corrupted key (accidental or intentional) from ever being used
- APIs to encrypt and decrypt data using secure keys
  - Use in conjunction with message digest APIs enables your applications to ensure data integrity (detect data corruption)
- Performance and archive advantages over external crypto box solutions
- To summarize, a solution that enables you to protect vital data, and does so with traditional TPF scalability and performance characteristics

Closing message - the choice is yours ...



with secure key management



without secure key management

Trademarks

IBM is a trademark of International Business Machines Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

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.