z/OS 3.2 IBM Education Assistant

Solution Name: System SSL PBKDF2 Enhancements (Enhanced PKCS#12, Enhanced Key and Request DB), In-memory private key protection and enhanced PKCS#7

Solution Element(s): System SSL

July 2025

Agenda

- Trademarks
- Objectives
- Overview
- Usage & Invocation
- Interactions & Dependencies
- Upgrade & Coexistence Considerations
- Installation & Configuration
- Summary
- Appendix

Trademarks

- See url http://www.ibm.com/legal/copytrade.shtml for a list of trademarks.
- Additional Trademarks:
  - None

Objectives

At the end of this presentation, you will have an overview and understanding of the following enhancements from System SSL in 3.2, 3.1 and 2.5:

- Provide options to build and read PKCS#12 files that are protected using PBES2 - PBKDF2 with AES
- Provide an option to create key databases (KDB) that are protected with PBES2 - PBKDF2 with AES
- Protect in-memory private keys with PBES2 - PBKDF2 with AES
- Provide options to use PBES2 - PBKDF2 with AES to build and read PKCS#7

Overview

- The common theme of all these new supports is based on the use of Password-Based Encryption Scheme (PBES) and Password-Based Key Derivation Function (PBKDF) with AES
- PKCS#12 (Public Key Cryptography Standard #12) is a standard for storing certificate(s) and a private key
- Security and integrity on this content are enforced by various algorithms playing together
  - For security:
    - the PKCS#12 standard uses a password-based encryption algorithm (PBES) combined with a key derivation function (PBKDF) to generate a symmetric encryption key such as TDES or AES, based on a password
  - For integrity:
    - the PKCS#12 standard ensures data integrity by employing hashing algorithms such as SHA1 or SHA256, to verify the content built by this standard
- Summary of the new supports
  - PBES1, PBKDF1, TDES and SHA1: before the enhancements
  - PBES1, PBKDF1, TDES and SHA1 + PBES2, PBKDF2, AES and SHA256/384/512: after the enhancements

Enhanced PKCS#12 Support

Overview

- Who (Audience)
  - z/OS customers who need to handle PKCS#12 packages using System SSL
- What (Solution)
  - With this enhancement, System SSL supports importing and exporting of PKCS#12 packages with algorithm PBES2-PBKDF2 with AES
- Wow (Benefit / Value, Need Addressed)
  - This enhancement implements the recommendation from NIST SP800-132 to move away from DES and TDES.
  - Provide the capability to **accept** a PKCS#12 package created with these newer algorithms from other products like OpenSSL, keytool…; and to **create** such a package to be consumed by other products
  - Note: PBKDF2 with SHA256/384/512 is considered quantum safe

Usage & Invocation (1)

- gskkyman export command (-e) for pkcs12 now has an encryption algorithm input option marked in blue (-kpw and –p12pw are only valid in z/OS 3.1 and above)
  - `gskkyman –e –k <kdb name> -l <cert label> -p <output p12 file name> -ealg <algorithm> [-kpw <kdb pw> or –sth] [-p12pw <p12 pw>]`
  - `gskkyman –e –t <pkcs11 token name> -l <cert label> -p <output p12 file name> -ealg <algorithm>`
- Valid -ealg values:
  - 3des (original support using PBES1 and TDES with SHA-1) – this is the default if –ealg is not specified
  - aes-sha256 (new support using PBES2 – PBKDF2 with 256-bit AES-CBC with SHA-256)
  - aes-sha384 (new support using PBES2 – PBKDF2 with 256-bit AES-CBC with SHA-384)
- When aes-sha256 or aes-sha384 is specified, the value for –p12pw must be at least 8 characters

Usage & Invocation (2)

- gskkyman interactive menu for pkcs#12 export with new options (in blue)

```
Export File Format

  1 - Binary PKCS #12 Version 1
  2 - Base64 PKCS #12 Version 1
  3 - Binary PKCS #12 Version 3
  4 - Base64 PKCS #12 Version 3

Select export format (press ENTER to return to menu): 3
Enter export file name (press ENTER to return to menu): myp12.p12
Enter export file password (press ENTER to return to menu):
Re-enter export file password:
Enter 2 for encryption list, 1 for strong encryption, 0 for export encryption: 2

Export File Encryption
  1 - Triple DES with SHA-1 digest
  2 - 256-bit AES-CBC with SHA-256
  3 - 256-bit AES-CBC with SHA-384
Select encryption (press ENTER to return to menu): 2
```

Usage & Invocation (3)

- APIs new support
  - **gsk_export_key()** and **gsk_encode_export_key()** input parameter algorithm now includes
    - x509_alg_pbes2WithSha256AndAesCbc256
      - 256-bit AES-CBC with SHA-256 digest and PBKDF2 HMAC SHA-256.
    - x509_alg_pbes2WithSha384AndAesCbc256
      - 256-bit AES-CBC with SHA-384 digest and PBKDF2 HMAC SHA-384
    - Note that the above options need a password with at least 8 characters
  - The APIs that handle the original PKCS#12 PBES1 algorithm have been updated to handle the above algorithms, e.g. gsk_import_key() and gsk_decode_import_key()

Interactions & Dependencies

- Software Dependencies
  - None
- Hardware Dependencies
  - None
- Exploiters
  - System SSL applications and end users

Upgrade & Coexistence Considerations

- To exploit this solution, all systems in the Plex must be at the new z/OS level:
  - No
- List any toleration/coexistence APARs/PTFs.
  - N/A
- List anything that doesn't work the same anymore.
  - N/A
- Upgrade involves only those actions required to make the new system behave as the old one did.
  - N/A
- Coexistence applies to lower-level systems which coexist (share resources) with latest z/OS systems.
  - N/A

Installation & Configuration

- Application needs to have access to the profiles in the CSFSERV class protecting the CSF1GSK, CSF1GAV and CSF1TRD resource

Enhanced Key and Request Database

Overview

- System SSL stores **certificates and keys** in a key database and stores **certificate requests** in a corresponding request database
- Both are implemented as files in the zFS file system with file extensions of .kdb and .rdb respectively
- When a key database is created, a request database will be created automatically with the same name with the .rdb extension
- The database is protected by a password which can be optionally stored in a stash file created with the same database name with the .sth extension
- A set of these 3 files are created under the same directory, eg:
  - /tlshome/ftpstore/sslstore.kdb
  - /tlshome/ftpstore/sslstore.rdb
  - /tlshome/ftpstore/sslstore.sth

Overview (2)

- Who (Audience)
  - z/OS customers who need the System SSL key and request databases to be protected with stronger algorithms
- What (Solution)
  - With this enhancement, System SSL provides options to create new key/request databases and convert current ones using algorithm PBES2-PBKDF2 with AES
  - The convert function also supports the conversion back to the original version, if needed
- Wow (Benefit / Value, Need Addressed)
  - This enhancement implements the recommendation from NIST SP800-132 to move away from DES/TDES.
  - Enable users to create new key/request databases and convert current ones with stronger algorithms.
  - Note: PBKDF2 with SHA256/384/512 is considered quantum safe

Usage & Invocation (1)

- gskkyman supports a new convert command (-c) to convert the original (version 1) key/request databases to stronger ones (version 2) or vice versa
  - `gskkyman -c`
    `-k <source kdb name> -kpw <source kdb pw> | -sth >`
    `-nk <target kdb name> -nkpw <target kdb pw> -v <target version 1|2>`

  - If the target version is 2, the current key/request databases (version 1) will be converted to a v2 version
  - If the target version is 1, the current key/request databases (version 2) will be converted to a v1 version
  - The password for the V2 database must be at least 8 characters
  - The source key/request/stash files (.kdb, .rdb, .sth) remain untouched
  - If a stash file for the target version is desired, you will need to create it with the –s function
    - `gskkyman -s  -k <kdb name> -kpw <kdb pw>`
    - The stash file would be created with the same name as the input kdb name with extension .sth
  - You may either rename the new files back to the original ones OR update the application to specify the new names

Usage & Invocation (2)

- gskkyman interactive menu for version 2 database support (new options in blue)

Database Menu
1 - Create new database
1b - Create new empty database
…

Original option always creates populated DB with some well- known CA certificates

Enter option number: 1
Enter key database name (press ENTER to return to menu): myv2store.kdb
Enter database password (press ENTER to return to menu):
Re-enter database password:
Enter password expiration in days (press ENTER for no expiration):
Enter database record length (press ENTER to use 5000):

Enter 2 for database types, 1 for FIPS mode database or 0 to continue: 2

Database Types

1 - non-FIPS
2 - FIPS
3 - non-FIPS Version 2 (empty)
4 - FIPS Version 2 (empty)

For version 2 DB, it is not populated, i.e. empty.

Usage & Invocation (3)

- gskkyman interactive menu for version 2 database support (new options in blue)

Database Menu
1 - Create new database
1b - Create new empty database
…
Enter option number: 1b
Enter key database name (press ENTER to return to menu): myv2store.kdb
Enter database password (press ENTER to return to menu):
Re-enter database password:
Enter password expiration in days (press ENTER for no expiration):
Enter database record length (press ENTER to use 5000):

Enter 2 for database types, 1 for FIPS mode database or 0 to continue: 2

Database Types

1 - non-FIPS
2 - FIPS
3 - non-FIPS Version 2
4 - FIPS Version 2

Usage & Invocation (4)

- APIs new support
  - **gsk_create_database()** input parameter db_type now includes
    - gskdb_dbtype_key_v2
      - Empty key database protected with PBES2-PBKDF2 with AES algorithm
    - gskdb_dbtype_request_v2
      - Certification request database protected with PBES2-PBKDF2 with AES algorithm
  - All the APIs that handle version 1 databases have been updated to support the version 2 ones, e.g.
    - gsk_create_database(), gsk_open_database(), gsk_create_database_signed_certificate(), gsk_export_certificate()…

Interactions & Dependencies

- Software Dependencies
  - None
- Hardware Dependencies
  - None
- Exploiters
  - System SSL applications and end users

Upgrade & Coexistence Considerations

- To exploit this solution, all systems in the Plex must be at the new z/OS release level (3.2) or install the corresponding PTFs for the older releases (2.5, 3.1)
  - Version 2 databases can’t be opened from the system that does not have this support
  - If the version 2 database is to be copied to the system that does not have this support, it must be converted to version 1 first by using the gskkyman convert function (gskkyman –c)
- List any toleration/coexistence APARs/PTFs.
  - N/A
- List anything that doesn't work the same anymore.
  - N/A
- Upgrade involves only those actions required to make the new system behave as the old one did.
  - N/A
- Coexistence applies to lower-level systems which coexist (share resources) with latest z/OS systems.
  - N/A

Installation & Configuration

- Application needs to have access to the profiles in the CSFSERV class protecting the CSF1GSK, CSF1GAV and CSF1TRD resource

In-memory private key protection and enhanced PKCS#7

Overview

- Who (Audience)
  - z/OS customers who want to secure in-memory keys read in from the SAF keyrings, z/OS PKCS#11 tokens, and PKCS#12 files and handle PKCS#7 data with PBES2-PBKDF2 with AES
- What (Solution)
  - With this enhancement, System SSL enables the protection of in-memory keys read from the SAF keyrings, z/OS PKCS#11 tokens, and PKCS#12 files, and the reading and building PKCS#7 data with algorithm PBES2-PBKDF2 with AES
- Wow (Benefit / Value, Need Addressed)
  - This enhancement implements the recommendation from NIST SP800-132 to move away from DES/TDES
  - Enhance the in-memory keys protection
  - Provide the capability to read and build PKCS#7 data with these newer algorithms
  - Note: PBKDF2 with SHA256/384/512 is considered quantum safe

Usage & Invocation (1)

- When System SSL retrieve the certificates and keys from the SAF keyrings, z/OS PKCS#11 tokens, and PKCS#12 files into memory, (e.g. during the handshake process), the keys are protected by PBES2-PBKDF2 with AES automatically when ICSF is available
- Using PBES2-PBKDF2 with AES on the following enhancements no longer need access to the profiles in the CSFSERV class protecting the CSF1GSK, CSF1GAV and CSF1TRD resources
  - PKCS#12 import and export
  - Version 2 key and request databases

Usage & Invocation (2)

- APIs new support
  - **gsk_make_encrypted_data_content()** and **gsk_make_encrypted_data_msg()** input parameter algorithm now includes
    - x509_alg_pbes2WithSha256AndAesCbc256
      - 256-bit AES-CBC with SHA-256 digest and PBKDF2 HMAC SHA-256.
    - x509_alg_pbes2WithSha384AndAesCbc256
      - 256-bit AES-CBC with SHA-384 digest and PBKDF2 HMAC SHA-384
  - **gsk_read_encrypted_data_content()** and **gsk_read_encrypted_data_msg()** can support the above algorithms

Interactions & Dependencies

- Software Dependencies
  - None
- Hardware Dependencies
  - None
- Exploiters
  - System SSL applications and end users

Upgrade & Coexistence Considerations

- To exploit this solution, all systems in the Plex must be at the new z/OS level:
  - No
- List any toleration/coexistence APARs/PTFs.
  - ICSF UJ95989 (HCR77E0 - 3.1)
  - ICSF UJ95990 (HCR77D2 - 2.5)
- List anything that doesn't work the same anymore.
  - N/A
- Upgrade involves only those actions required to make the new system behave as the old one did.
  - N/A
- Coexistence applies to lower-level systems which coexist (share resources) with latest z/OS systems.
  - N/A

Installation & Configuration

- The requirement to enable the application access to the profiles in the CSFSERV class protecting the CSF1GSK, CSF1GAV and CSF1TRD resource is lifted. (Note: still need ICSF to be up and running)

Summary

You should now be able to understand the following enhancements from System SSL in 3.2, 3.1 and 2.5:

- Provide options to build and read PKCS#12 files using PBES2 - PBKDF2 with AES
- Provide an option to create key databases (KDB) with PBES2 - PBKDF2 with AES
- Protect in-memory private keys with PBES2 - PBKDF2 with AES
- Provide options to use PBES2 - PBKDF2 with AES to build and read PKCS#7

Appendix

- Publication
  - z/OS Cryptographic Services System Secure Sockets Layer Programming