Feature Abstract by First Appearance in SDK – Security Features

Abstract	SDK 6.0.0	SDK 6.0.1	SDK 7
Mixed-case Keylabel Alias Support	GA	GA	GA
PKCS11Impl provider on z/OS	GA	GA	GA
4096-bit RSA keys	SR1	GA	GA
Key wrapping of AES keys with TDES master keys	SR1	GA	GA
KeyTool 6.0, -all	SR1	GA	GA
Support Self Signed Certificates in RACF	SR1	GA	GA
Optionally use MD5 in IBMJCECCA	SR2	GA	GA
CP Assist AES-256 Support	SR3	GA	GA
Random Number Generator Long Enhancements	SR3	GA	GA

Feature Abstract by First Appearance in SDK – Security Features (cont)

Abstract	SDK 6.0.0	SDK 6.0.1	SDK 7
Hardware crypto exploitation for SHA/256, SHA/384, and SHA/512	SR3	GA	GA
Remove dependency on unrestricted policy files for IBMJCECCA	SR7	GA	GA
Delete keys from PKDS via label	SR7	GA	GA
Key wrapping of CKDS label clear keys with RSA	SR7	GA	GA
Support PCICC Keys in RACF-based keystores	SR7	GA	GA
RAS - Enhanced ICSF Exception Handling	N/A	GA	GA
Support for AES Protected Keys	N/A	GA	GA
Secure Random Number Generator Caching Support	N/A	SR4	SR3

Feature Abstract by First Appearance in SDK – Security Features (cont)

Abstract	SDK 6.0.0	SDK 6.0.1	SDK 7
Add ECC support to IBMJCECCA security provider	N/A	N/A	SR4
Support 64-bit ICSF interfaces for all of IBMJCECCA	N/A	N/A	SR1
Hwkeytool Support for PKCS#11 Keystores	N/A	N/A	SR1
IBMJCE/IBMJCECCA RACF Converged Keystore	N/A	N/A	SR1
IBMJCEHybrid Provider	N/A	N/A	SR1
DES & AES Cipher mode CFB-LCFB and OFB	N/A	N/A	SR3
GCM mode for AES cipher	N/A	N/A	SR3
CBC Wrapping Mode	N/A	N/A	SR3
PKA RSA OAEP with SHA-256	N/A	N/A	SR3
Addition of ECC support to hwkeytool	N/A	N/A	SR5
Enablement of ECC support in IBMPKCS11Impl security provider	N/A	N/A	SR5

Feature Abstract by First Appearance in SDK – Security Features (cont)

Abstract	SDK 7.0	SDK V7R1	SDK V8.0
Enablement of ECC support in IBMPKCS11Impl security provider	SR5	GA	GA
IBMPKCS11Impl support for AES GCM (h/w)	SR6	GA	GA
Enterprise PKCS#11 Hardware Security Module	N/A	GA	GA
IBMJCECCA: Upgrade keystore protection from TDES to AES	SR8FP10	SR2FP10	GA
Enable PKCS#11 provider for use with JSSE2	SR8FP10	SR2FP10	GA
Key Encrypting Key support in IBMJCECCA	N/A	SR3FP20	SR2
Support ANSI TR-31 in IBMJCECCA	N/A	SR3FP40	SR3
IBMJCECCA h/w cryptographic provider for Linux on z Systems	N/A	N/A	SR3FP10
Secure HMAC support in IBMJCECCA	N/A	SR3FP60	SR3FP20

Feature Abstract by First Appearance in SDK – Security Features (cont)

Abstract	SDK 7.0	SDK V7R1	SDK V8.0
Addition of **PlatformAccessControl.checkMyPermission**	N/A	N/A	SR4FP2

Java Security – Feature Descriptions (order matches order of appearance in tables)

- Mixed-case Keylabel Alias Support - Java implementations have always created and used key aliases in lower-case form. TSO-based utilities permit creation of RACF key aliases with mixed-case labels, so it is possible to create RACF keys through this interface which cannot be referenced using standard Java APIs. This feature provides a means to use keys which were created with mixed-case (which includes upper-case) label aliases.

- PKCS11Impl provider on z/OS - Enablement of IBMPKCS11Impl provider on z/OS to exploit PKCS#11 services introduced in z/OS V1.9.

- 4096-bit RSA keys - Enhanced the IBMJCECCA provider to generate 4096-bit RSA key pairs and to perform encryption using 4096-bit RSA keys. 4096-bit is the largest RSA keysize supported by hardware crypto devices at this time.

- Key wrapping of AES keys with TDES master keys - This feature adds support to IBMJCECCA security provider to allow for the generation of secure AES keys wrapped by a TDES master key.

- KeyTool 6.0, -all - Enhanced hwkeytool utility program to be functionally equivalent to the software-based keytool utility program at the Java 6 level, including new or revised commands changealias, exportseckey, and importseckey. Includes addition of the -all option for the keypasswd and storepasswd commands.

- Support Self Signed Certificates in RACF - In early versions of the IBMJCECCA security provider, self signed certificates were not supported within a RACF keyring. This enhancement permits use of a self signed certificate on a RACF keyring without the requirement of having a random CA certificate also connected.

Java Security – Feature Descriptions (order matches order of appearance in tables)

- Optionally use MD5 in IBMJCECCA - Added support to IBMJCECCA security provider to set a Java environment variable to have IBMJCECCA use IBMJCE for MD5 hashing operations, for instances where the software implementation proves to be faster than the hardware implementation (due to instruction pathlength associated with h/w crypto support).
- CP Assist AES-256 Support - Update to IBMJCECCA provider to perform AES 256 operations via calls directly to assembler instructions wherever such support is available on CP Assist processor.
- Random Number Generator Long Enhancements - Enhancement to IBMJCECCA security provider to exploit the new Random Number Generator Long (RNGL) capability in hardware crypto firmware to permit fetching up to 8192 bytes at a time.
- Hardware crypto exploitation for SHA/256, SHA/384, and SHA/512 - The System z10 EC introduced additional cryptographic hardware to support SHA/256, SHA/384, and SHA/512 for KeyGenerator, Mac, MessageDigest, and Signature service types. This feature provides support in IBMJCECCA crypto provider for these algorithms executed in crypto hardware.
- Remove dependency on unrestricted policy files for IBMJCECCA - Many customers utilize key sizes that do not require the Java unrestricted policy files. However, prior to this feature, with the IBMJCECCA provider, any usage of the Java Cipher APIs to encrypt, decrypt, wrap, and unwrap using RSA private keys of any size will cause the API to fail unless the Java unrestricted policy files are installed. This support is intended to remove this restriction and therefore reduce the configuration burden.
- Delete keys from PKDS via label - Until this feature was introduced, if an asymmetric key object was deleted from a keystore by a user application using the keystore APIs, and if the key object had a PKDS label, the PKDS information associated with the deleted key was not deleted from the PKDS. This feature enhanced IBMJCECCA to provide an API to enable deletion of a PKDS entry for an asymmetric(RSA, DSA) key object.

Java Security – Feature Descriptions
(order matches order of appearance in tables)

- Key wrapping of CKDS label clear keys with RSA - This feature adds support to IBMJCECCA security provider to wrap hardware-based clear symmetric encryption keys (thereby creating key-encrypting keys) using RSA algorithm, thereby providing industry-standard support for wrapping key materials for transport using public key resources.
- Support PCICC Keys in RACF-based keystores - In early versions of the IBMJCECCA security provider, PCICC private keys were not supported when using a RACF keystore with the provider. However, use of the ICSF parameter on the RACF RACDCERT command enabled the IBMJCECCA provider to support them. This enhancement added support to the IBMJCECCA provider to use PCICC private keys in a RACF keystore.
- RAS - Enhanced ICSF Exception Handling - Added a new run time exception handler class to IBMJCECCA security provider to allow an application to programmatically retrieve either the API name of the service call, the message string, the return code or the reason code from an exception message thrown by the provider in response to a failed ICSF services call.
- Support for AES Protected Keys - Enhanced IBMJCECCA security provider to support AES protected keys (keys encrypted using the host master key). Prior to this feature, only AES clear keys were supported by IBMJCECCA.
- Secure Random Number Generator Caching Support - This feature provides a random number cache solution in the IBMJCECCA security provider which reduces the per-call CCA coprocessor overhead on the application's critical path while providing coprocessor quality random numbers for the hardware crypto (ICSF) calls. This caching implementation will cache random numbers that are generated by ICSF services, which MAY provide greater performance in scenarios frequently generating small sets of random numbers.

Java Security – Feature Descriptions (order matches order of appearance in tables)

- Add ECC support to IBMJCECCA security provider - Elliptic curve cryptography (ECC) is an encryption system that uses the properties of elliptic curves to provide the same functionality as other public cryptosystems such as encryption/decryption, key agreement and digital signatures. This feature adds these capabilities to IBMJCECCA to exploit these capabilities on the Crypto Express 3 Coprocessor. Updates to hwkeytool include include generating ECC key pairs, saving and loading to keystores (JCECCAKS and JCECCARACFKS), and exporting and importing certificates to and from keystores.
- Support 64-bit ICSF interfaces for all of IBMJCECCA - Enhancement to IBMJCECCA security provider to support three ICSF API's in 64 bit mode which were previously supported only in 31 bit mode when initially implemented in the provider. Includes addition of Key Record Create, Key Record Write, and PKDS Record Delete.
- Hwkeytool Support for PKCS#11 Keystores – Feature enabled PKCS#11 features in hwkeytool utility program for use with PKCS11Impl provider on z/OS.
- IBMJCE/IBMJCECCA RACF Converged Keystore - A new keystore has been created to accompany the IBMJCEHYBRID provider. This keystore will allow a user to load a single SAF keyring as both a JCECCARACFKS and JCERACFKS keystore representation. When loaded, the new JCEHYBRIDRACFKS keystore contains a mashup of the key and certificate material from both the JCECCARACFKS and JCERACFKS keystore representations of the corresponding SAF keyring. This key container contains all of the available key material from both keystores.
- IBMJCEHybrid Provider - Prior to this feature, exploiters if IBMJCECCA security provider would need to explicitly code applications to make transition to JCE software when h/w crypto failure exceptions arose in IBMJCECCA. This feature makes that transition automatic with no application changes.

Java Security – Feature Descriptions (order matches order of appearance in tables)

- DES & AES Cipher mode CFB-LCFB and OFB - z/OS V1R12 introduced additional modes for the DES and AES ciphers. This feature enhances IBMJCECCA crypto provider to support those cipher modes.
- GCM mode for AES cipher - Exploits new hardware cryptographic support added on zSeries cryptographic processors.
- CBC Wrapping Mode - The ANSI X9.24 standard (Retail Financial Services Symmetric Key Management) requires the key value in a key token be bundled with other token data and encrypted using cipher block chaining (CBC) mode. Enhanced support for Java services in the IBMJCECCA cryptographic provider to generate and import key tokens that are X9.24 compliant.
- PKA RSA OAEP with SHA-256 - RSA OAEP (or RSA Encryption Scheme – Optimal Asymmetric Encryption Padding) is a public-key encryption scheme for encoding in combination with the RSA algorithm and a hash algorithm. Previous to this enhancement IBMJCECCA supported RSA OAEP with a SHA-1 algorithm as defined by the PKCS #1 V2.0 standard. PKCS #1 V2.1 extended the OAEP method to also allow SHA-256 hashing. This enhancement permits the use of either SHA-1 or SHA-256 to meet that new standard.
- Addition of ECC support to hwkeytool - Elliptic curve cryptography (ECC) is an encryption system that uses the properties of elliptic curves to provide the same functionality as other public cryptosystems such as encryption/decryption, key agreement and digital signatures. This feature adds updates to hwkeytool including generating ECC key pairs, saving and loading to keystores (JCECCAKS and JCECCARACFKS), and exporting and importing certificates to and from keystores.
- Enablement of ECC support in IBMPKCS11Impl security provider – Similar to **Add ECC support to IBMJCECCA security provider**, this feature enables Elliptic curve cryptography support in the IBMPKCS11Impl security provider on z/OS.

Java Security – Feature Descriptions (order matches order of appearance in tables)

- IBMPKCS11Impl support for AES GCM (h/w) – Adds support for GCM mode in AES algorithm to exploit hardware crypto through the IBMPKCS11Impl provider.
- Enterprise PKCS#11 Hardware Security Module – Exploitation of ICSF Enterprise PKCS#11 secure key support in the Token Key Data Set (TKDS), including RACF references to those secure keys, via the PKCS11Impl provider. “Secure Key” means that sensitive key material is always wrapped under a host master key.
- IBMJCECCA Upgraded keystore protection from TDES to AES - The content of JCECCAKS Java key stores created in Java 7 SR 8 FP10, Java 7.1 SR 2 FP10, or Java 8 (and in future releases) is protected with AES encryption, replacing the TDES (DESede) encryption used in previous Java releases. Keystores and key entries created at earlier service levels continue to be supported. However, a keystore created with the new, stronger protection is not downward compatible with earlier maintenance levels.
- Enable PKCS#11 provider for use with JSSE2 – In addition to using the IBMJCECCA hardware cryptographic provider with the IBMJSSE2 provider to achieve encryption of handshake and payload data, this feature enables use of the IBMPKCS11Impl hardware cryptographic provider for the same capabilities.
- Key Encrypting Key support in IBMJCECCA - Key encrypting keys are used to safely transfer hardware keys between two systems. This basic symmetric key encrypting key support consists of symmetric key generation and Diffie-Hellman key derivation of both DESede and AES key encrypting keys and hardware key wrapping and unwrapping of DES, DESede and AES keys using key encrypting keys.
- Support ANSI TR-31 in IBMJCECCA - Various vendors use different proprietary schemes for encoding symmetric key attributes on different platforms. In an enterprise where there is a need to exchange key materials across dissimilar platforms, TR-31 provides a standardized way in which customers can securely exchange key material and its attributes between these differing cryptographic systems.

Java Security – Feature Descriptions (order matches order of appearance in tables)

- IBMJCECCA h/w cryptographic provider for Linux on z Systems – The IBMJCECCA cryptographic provider which offers support for cryptographic operations through exploitation of hardware cryptographic devices using Common Cryptographic Architecture (CCA) interfaces is shipped with the Java SDK for Linux on z Systems. It contains comparable function to the IBMJCECCA provider on z/OS, where available on Linux.
- Secure HMAC support in IBMJCECCA - The IBMJCECCA provider now supports HMACSHA1, HMACSHA224, HMACSHA256, HMACSHA384, and HMACSHA512 SecretKeyFactory algorithms for creating a Java key object from a key specification that contains the label of an existing CKDS secure HMAC key entry. The hwkeytool command line utility is extended to create an HMAC Java™ key object (com.ibm.crypto.hdwrCCA.provider.HMACKey) from an existing CKDS entry that contains a secure HMAC key, then store that key object in the IBMJCECCA keystore (JCECCAKS).
- Addition of **PlatformAccessControl.checkMyPermission** - This access control method is used to check whether a specific user has permission to a resource. A similar method, checkPermission, requires the current user to either have READ access to BPX.SERVER or to have superuser status. However, checkMyPermission() only requires that the current user has a task-level Access Control Environment Element (ACEE).

Feature Abstract by First Appearance in SDK – Legacy Features

Abstract	SDK 6.0.0	SDK 6.0.1	SDK 7
Introduction of JZOS to SDK	SR1	GA	GA
JZOS SMF record write support	SR1	GA	GA
JZOS Rauditx support	SR1	GA	GA
JZOS additional system services	SR3	GA	GA
JZOS implement IDCAMS	SR3	GA	GA
z/OS DFSORT API Support	SR3	GA	GA
Add support to JZOS for reading Extended Address Volume (EAV) Data Set Control Blocks (DSCBs)	N/A	GA	GA
Add support to JZOS for z/OS logstream read/browse	N/A	GA	GA
Add support to JZOS for native job submission	N/A	GA	GA

Feature Abstract by First Appearance in SDK – Legacy Features (cont)

Abstract	SDK 6.0.0	SDK 6.0.1	SDK 7
JZOS Add support for Workload Manager (WLM) services	N/A	GA	GA
JZOS Add IO interfaces for z/OS logstreams	N/A	GA	GA
JRIO Deprecation and migration to JZOS	N/A	GA	GA
JZOS ZFile Enqueue Support	N/A	GA	GA
JZOS Sequential Dataset I/O Performance Improvement	N/A	SR1	SR1
JZOS: zCompression wrapper (CMPSC compression wrapper)	N/A	N/A	SR3

Feature Abstract by First Appearance in SDK – Legacy Features (cont)

Abstract	SDK 7.0	SDK V7R1	SDK V8.0
JZOS WLM Query Contention wrapper	N/A	GA	GA
Java Data Access Accelerator – JZOS exploitation	N/A	GA	GA
Ability to force an abnormal termination in JZOS	N/A	GA	GA
JZOS API to support USS file tagging in Java	N/A	SR2FP10	GA
JZOS JES Symbol Support and Preliminary Support for User Job Correlators	N/A	SR2FP10	GA
Provide JMX based performance logging to SMF for the JZOS launcher	N/A	SR2FP10	GA
Removal of JRIO component from z/OS Java SDK	N/A	N/A	GA
Augment JZOS launcher SMF logging data with CPU consumption data	N/A	SR3FP20	SR1
Augment JZOS launcher SMF logging data with native thread ID	N/A	SR3FP20	SR2

Java Batch – Legacy Feature Descriptions (order matches order of appearance in tables)

- Introduction of JZOS to SDK – An aggregation of legacy services support through Java was first introduced into the SDK at this time.
- JZOS SMF record write support – Feature provided Java APIs for z/OS SMF record writing for applications needing to create the z/OS SMF log records from a Java application.
- JZOS Rauditx support – Feature provided Java APIs which enable Java applications to record security relevant events to the z/OS System Management Facility (SMF) repository. Specifically, this API enables the creation of security SMF 83 type 5 records that enable Java applications to write SMF 83 records in a controlled fashion.
- JZOS additional system services - Feature provided additional system services to Java on z/OS that allow customers to: 1) use currently available system serialization on z/OS resources via ENQ/DEQ (ISGENQ) , 2) access system-wide symbols, 3) provide new services for getting clock values using STCK and STCKE instructions, and 4) redirect all output streams into a single DD output statement.
- JZOS implement IDCAMS – Feature provided a Java API to allow applications to use the currently available system utility functions for creating and deleting VSAM datasets.
- z/OS DFSORT API Support – Feature provided a Java API to spawn a child process to run an executable program that invokes DFSORT, and which supports multiple concurrent requests.
- Add support to JZOS for reading Extended Address Volume (EAV) Data Set Control Blocks (DSCBs) – Prior to this feature, the JZOS API for the OBTAIN Macro returned only Format-1 DSCB information. This feature enhanced JZOS to provide support for extended access volumes (Format-8 and Format-9 DSCBs), introduced in z/OS V1.10.

Java Batch – Legacy Feature Descriptions (order matches order of appearance in tables)

- Add support to JZOS for z/OS logstream read/browse – Feature provided an API to read existing logstreams from Java (in addition to being able to write to logstreams, function which already existed in prior versions of the SDK).
- Add support to JZOS for native job submission – Feature provided an API in JZOS for submitting jobs to the internal reader and returning the Job ID to the caller.
- Add support for Workload Manager (WLM) services - Java applications all run against the default RMF profile. This feature added a set of Java APIs to create / join / leave / delete a WLM work unit which allow Java processes to be tracked discretely by WLM.
- Add IO interfaces for z/OS logstreams – Feature enabled z/OS Logstreams to be writable using standard Java interfaces (java.io.OutputStream).
- JRIO Deprecation and migration to JZOS - The record I/O support in the JZOS component is designated as the strategic implementation, supplanting JRIO in future SDKs. JRIO has been available and widely used since being introduced in SDK1.4.2 SR6. In SDK 7, JRIO classes are marked deprecated and generate compiler warnings for customers using these classes, indicating that these classes are no longer considered strategic implementations.
- JZOS ZFile Enqueue Support – Feature provided a new ZFile constructor that has the same capabilities as the existing ZFile constructor with two additional flags, FLAG_DISP_SHR and FLAG_PDS_ENQ, to automate the serialization support of PDS members (previously, this could be done explicitly using JZOS ENQ/DEQ support).

Java Batch – Legacy Feature Descriptions (order matches order of appearance in tables)

- JZOS Sequential Dataset I/O Performance Improvement – Feature implemented peformance improvements on BSAM reads and writes such that dataset I/O through JZOS now more closely approximates the performance of similar COBOL I/O to these datasets.
- JZOS: zCompression wrapper (CMPSC compression wrapper) – Feature provides Java APIs for invocation of compression function using the z/Architecture CMPSC hardware Compression instruction, rather than using software-based compression. The APIs support CMPSC, GZIP (from java.util.zip), and INFLATE/DEFLATE (from java.util.zip).
- JZOS WLM Query Contention wrapper - Adds support for Workload Manager (WLM) services by providing a Java binding for the SYSEVENT QRYCONT macro which debuts in z/OS V2R1.
- Java Data Access Accelerator – JZOS exploitation - SDK 7.1 introduced a Data Access Accelerator solution which includes a Java class library that provides a set of primitive data access/manipulation functions and JIT compiler which acts in concert with the Data Access Accelerator library, replacing known methods with generated inline hardware instructions when compiling Java methods that include calls to these special functions. The JZOS component integrates exploitation of DAA into the COBOL copy book record parsing infrastructure.
- Ability to force an abnormal termination in JZOS – Prior to SDK 7.1, when an abend occurs in a JZOS Batch Launcher job, JZOS exception handling takes effect and sets a return code greater than zero for the termination of the step. In such error situations, the normal JES mechanisms should take effect in the current step (dataset disposition). Previously it was possible to set a non-zero return code, but a non-zero return code is not an “abnormal termination” error situation for JES, and job step failure mechanisms were not invoked. A new optional environment variable for the JZOS Batch Launcher is available to force the Batch Launcher into abnormal termination. From an application design perspective, this new behavior of JZOS will match the traditional batch approach/principles.

Java Batch – Legacy Feature Descriptions (order matches order of appearance in tables)

- JZOS API to support USS file tagging in Java - Added new APIs for accessing z/OS USS specific file attributes, such as extended attribute flags (Shared Library, No Shareas, APF Authorized, Program Controlled), user and auditor audit flags, change time and reference time, file format, file tagging, and security label.
- JZOS JES Symbol Support and Preliminary Support for User Job Correlators - The JES Symbol Service, introduced in the z/OS V2R1 release, provides a single view of JCL and JES symbols. The service allows applications to create, update, delete, or extract JES symbols. This JZOS feature adds JES Symbol support APIs to enable applications to create, update, delete, and extract JES symbols.
- Provide JMX based performance logging to SMF for the JZOS launcher – Enables JZOS batch launcher to write a z/OS SMF record containing z/OS Java runtime performance information before JVM shutdown using existing JMX beans capabilities. The z/OS Java runtime performance information to be written to the SMF record includes JVM name, JVM creation time, JVM uptime, JVM Garbage Collection policy and Garbage Collector details, Peak thread count, and live thread count.
- Removal of JRIO component from z/OS Java SDK - The JRIO component of z/OS Java SDKs was deprecated in SDK V6.0.1. In accordance with the Statement of Direction in the SDK V7.0 Announcement, this component has been withdrawn from SDK V8. For SDK V8 and later releases, use the record I/O facilities in the JZOS component in lieu of the JRIO facilities.
- Augment JZOS launcher SMF logging data with CPU consumption data, native thread ID - SMF record 121.1 updated to add precise and accountable CPU resource consumption information for each of these four thread categories: Application, System-JVM (including GC and JIT). GC, JIT. In addition to summary information, individual per-thread detail may be optionally configured to include Java thread ID (including native z/OS thread ID), Thread name, Thread category, Total CPU time.