

# z/OS V1R9 ICSF PKCS#11 Support



Prepared by Patrick Kappeler  
IBM Consulting IT Specialist  
kappeler@fr.ibm.com

**Redbooks**  
International Technical Support Organization

© 2007 IBM Corporation

z Security Update

## Trademarks

See url <http://www.ibm.com/legal/copytrade.shtml> for a list of trademarks

### The following are trademarks or registered trademarks of other companies.

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

\* All other products may be trademarks or registered trademarks of their respective companies.

#### 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.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

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.

## Acronyms

▪AES	▪Advanced Encryption Standard	▪MAC	▪Message Authentication Code
▪ARL	▪Authority Revocation List	▪MDC	▪Message Detection Code
▪CA	▪Certification Authority	▪MD5	▪Message Digest 5
▪CBC	▪Cipher Block Chaining	▪OAEP	▪Optimal Asymmetric Encryption Padding
▪CCA	▪IBM Common Cryptographic Architecture	▪OCSF	▪OS/390 Open Cryptographic Services Facility
▪CCF	▪Cryptographic Coprocessor Facility	▪OCSP	▪Online Certificate Status Protocol
▪CDSA	▪Common Data Security Architecture	▪PCICA	▪PCI Cryptographic Accelerator
▪CEX2A	▪Crypto Express 2 Accelerator	▪PCICC	▪PCI Cryptographic Coprocessor
▪CEX2C	▪Crypto Express 2 Coprocessor	▪PCIXCC	▪PCIX Cryptographic Coprocessor
▪CFB	▪Cipher FeedBack	▪PKA	▪Public Key Architecture
▪CKDS	▪Cryptographic Key Data Set	▪PKCS	▪Public Key Cryptographic Standards
▪CRL	▪Certificate Revocation List	▪PKDS	▪Public Key Data Set
▪CRT	▪Chinese Remainder Theorem	▪PKI	▪Public Key Infrastructure
▪CVC	▪Card Verification Code	▪RA	▪Registration Authority
▪CVV	▪Card Verification Value	▪RACF	▪Resource Access Control Facility
▪DES	▪Data Encryption Standard	▪RSA	▪Rivest-Shamir-Adleman
▪DSA	▪Digital Signature Algorithm	▪SET	▪Secure Electronic Transaction
▪DSS	▪Digital Signature Standard	▪SHA-1	▪Secure Hash Algorithm 1
▪ECB	▪Electronic Code Book	▪SLE	▪Session Level Encryption
▪FIPS	▪Federal Information Processing Standards	▪SSL	▪Secure Sockets Layer
▪GSS	▪Generalized Security Services	▪TKE	▪Trusted Key Entry
▪ICSF	▪Integrated Cryptographic Service Facility	▪TLS	▪Transport Layer Security
▪IETF	▪Internet Engineering Task Force	▪VPN	▪Virtual Private Network
▪IPKI	▪Internet Public Key Infrastructure		
▪KGUP	▪Key Generation Utility Program		
▪LDAP	▪Lightweight Directory Access Protocol		

## Agenda

- Public Key Cryptography Standards – PKCS#11
- z/OS PKCS#11 Implementation
- PKCS#11 Support - RACF RACDCERT And R-datalib
- PKCS#11 Support - System SSL gskkyman



# Public Key Cryptography Standard PKCS#11

## Public Key Cryptography Standard # 11

<http://www.rsa.com/rsalabs/node.asp?id=2133>

### PKCS #11

- An platform independent cryptographic application programming interface, developed by RSA Laboratories
- Smart card interfacing standard - Also known as Cryptoki (Cryptographic Token Interface)
- A de facto industry standard on many computing platforms today

### PKCS#11 vs IBM CCA

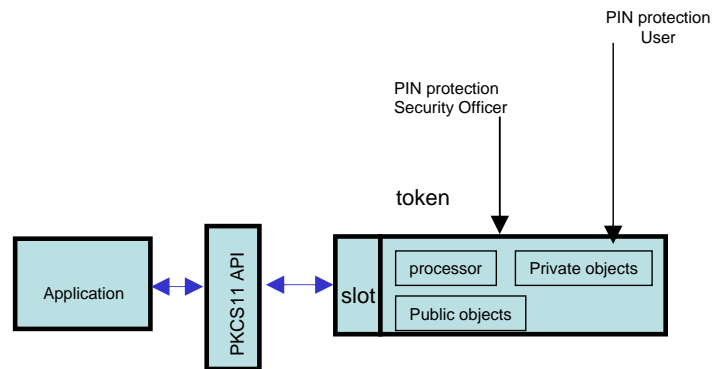
- Higher level API than CCA – Easier to use by C based applications
- As with IBM CCA, the persistent storage/retrieval of objects is part of the standard.
- Objects are certificates, keys, and even application specific data objects

### Subset of PKCS#11 API shipped in z/OS V1R9 for C applications

- Exploit hardware crypto through ICSF services
- Backed-up by changes in
  - ICSF
  - RACF
  - System SSL
- New ICSF book: Writing PKCS #11 Applications (SA23-2231)

## PKCS#11 implementation model

- **Token** – Logical view of a crypto device, e.g., smart card
- **Slot** – Logical view of a card reader, numbered 0 - n
- **Object** – Item stored on token, e.g., certificate, key, etc
- **User** – Owns the private data on a token by knowing the PIN
- **Security Officer (SO)** – Person who initializes a token



Load platform specific DLL  
 Call C\_GetFunctionList() – Returns addresses of C\_ functions  
 Call C\_Initialize() – Prepares address space for PKCS#11  
 Call C\_OpenSession() – Enables communication with a slot (token)  
 Call C\_Login() – Login to card with PIN as user or SO

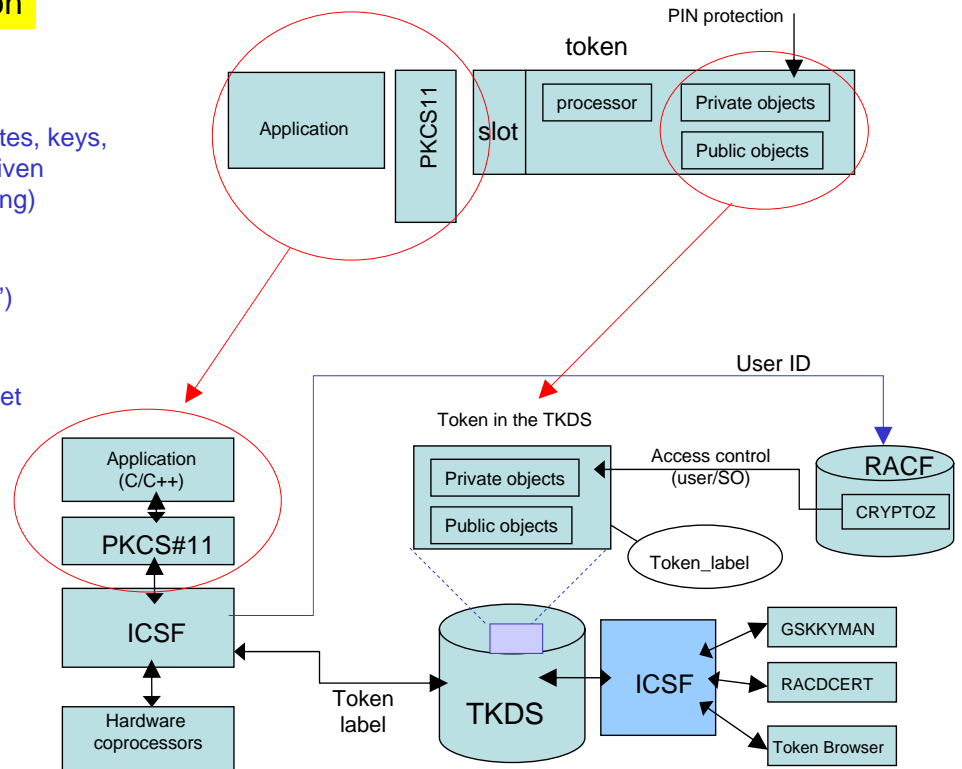
Call crypto functions as needed  
 e.g., C\_GenerateKeyPair(), C\_CreateObject(), C\_Encrypt()...

Call C\_Logout() -  
 Call C\_CloseSession()  
 Call C\_Finalize()

**z/OS PKCS#11  
Implementation**

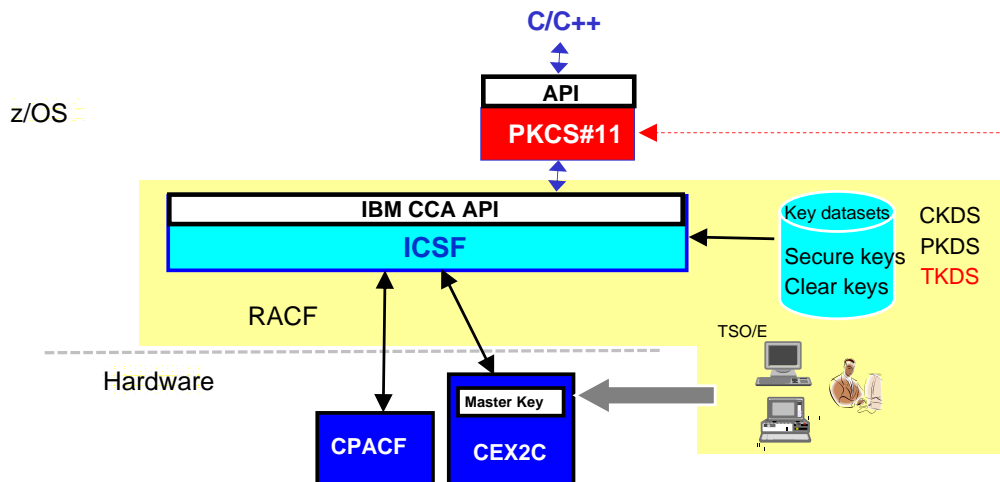
## z/OS PKCS#11 Implementation

- Tokens are virtual smart cards
  - Each is a collection of certificates, keys, data objects as needed by a given application (like a RACF key ring)
  - Token names (labels) - up to 32 chars (A-Z, 0-9, '@', '#', '\$', '.')
  - Tokens are kept in the TKDS (Token Data Set) VSAM dataset
  - Token access controlled by SAF profiles, not PINs
- Applications transparently share the hardware coprocessors (no direct access to cards)



## ICSF PKCS#11 Support - Introduction

- 31-bit, 31-bit XPLINK, 64-bit DLLs and sidedecks shipped in SYS1.SIEALNKE
- UNIX versions also shipped in /usr/lib and /usr/lpp/pkcs11/lib
- csnpdefs.h shipped in SYS1.SIEAHDR.H and /usr/include
- Sample code and makefiles shipped in /usr/lpp/pkcs11/samples



## z/OS PKCS#11 Implementation Summary

- **“C” Application Programming Interface**
  - Subset of the “C” functions defined in the PKCS #11 v2.20 specification (see appendix for installation)
  - See “z/OS Cryptographic Services Integrated Cryptographic Services Facility Writing PKCS #11 Applications” - SA23-2231
- **New subcomponents in ICSF**
  - Token Key Data Set (TKDS)
  - 5 new ICSF callable services for crypto and token management (see appendix)
  - Changed and new ISPF panels
- **RACF supports for PKCS #11**
  - The new CRYPTOZ class of profiles to protect tokens
  - Token and certificate management using the RACDCERT command
  - Token access through SAF key ring services
- **System SSL support for PKCS #11**
  - Token and certificate management using gskkyman

## z/OS PKCS#11 Implementation – Objects in the PKCS#11 token

### 5 object classes supported in the API

Each object has its own set of attributes with default values modifiable by applications

e.g. objects can have the private or public attribute that dictates which privilege is required to use it

#### Data

Default value: private

#### Secret key

default: private  
DES, DES2, DES3, AES

#### Certificate

default: public  
X.509 only

#### Public key

Default: public  
RSA only

#### Private key

default: private  
RSA only

## New ICSF ISPF Panel

## ICSF Token « browser » (ICSF Utility – Option 7)

```

----- ICSF Token Management - Token Details --- Row 1 to 1 of 5

Token name: HTTP.SERVER.TOKEN
Manufacturer: RACF HRF7740
Model: HCR7740
Serial Number: 0
Number of objects: 5

Select objects to process then press ENTER

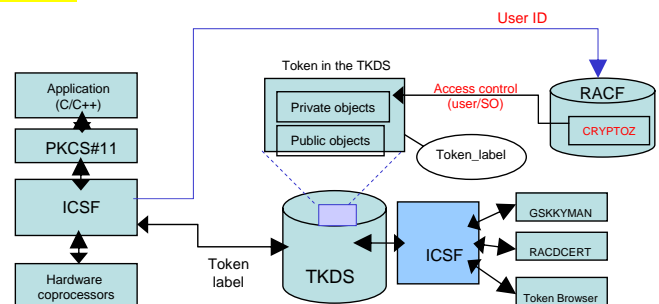
Press END to return to the previous menu.

-----
_ Object 1          CERTIFICATE    PRIVATE: FALSE    MODIFIABLE: TRUE
                   DEFAULT: FALSE    CATEGORY: Authority
  LABEL:           Widgets CA
  SUBJECT:         OU=Widgets CA, O=Widgets, Inc, C=US
  ID:              D5F4989FB6B622DCB6C8550F68EBE79651B1D75
  ISSUER:          OU=Widgets CA, O=Widgets, Inc, C=US
  SERIAL NUMBER:   00

COMMAND ==>>>                                SCROLL ==>> PAGE

```

## Token access control with the CRYPTOZ Class



The profile name indicates whether the permissions apply to the token user or the Security Officer (SO)

SO.<token\_label>  
USER.<token\_label>

The access level in the profile specifies the access type to the token

- Whether private or public objects can be accessed
- Whether the objects can be read only or created/modified
- 

The access types in z/OS PKCS#11 are

- User R/W, SO R/W, User R/O  
These are kept from the original PKCS#11 specifications
- Weak SO, Weak User, Strong SO  
These are specific to the z/OS implementation

See next slide

## Token access control with the CRYPTOZ Class

CRYPTOZ Resource	SAF Access Level		
	READ	UPDATE	CONTROL
<i>SO.token-label</i>	<b>Weak SO</b> - read / create / delete / modify / use public objects	<b>SO R/W</b> - Weak SO plus create / delete tokens	<b>Strong SO</b> - SO RW plus read (but not use) private objects, create / delete / modify private objects
<i>USER.token-label</i>	<b>User R/O</b> - read / use public and private objects.	<b>Weak User</b> - User R/O plus create / delete / modify private and public objects (cannot add / delete / modify certificate authority objects)	<b>User R/W</b> - Weak User plus add / delete / modify certificate authority objects

**Weak Security Officer** - can define the trust policy for a token (trusted CA's), but cannot initialize tokens. E.g. a corporate trust policy officer/auditor.

**Strong Security Officer** - can initialize tokens and populate them, but not use the keys. E.g. an application administrator.

**Weak User** - has access to everything in the token, but cannot alter the trust policy of the token. E.g. a server daemon



## PKCS#11 Tokens Management





## PKCS#11 Tokens Management



## RACF RACDCERT And R\_datalib

## RACF PKCS#11 Token Management Support

- New RACDCERT token functions can manage tokens in the TKDS in a manner similar to RACF keyrings  
Requires ICSF up and running
  - **RACDCERT ADDTOKEN** - Creates an empty token with a given name
  - **DELTOKEN** - Destroys the token if it exists
  - **BIND** - Adds a RACF certificate to a token
  - **UNBIND** - Removes a certificate from a token
  - **LISTTOKEN** - Lists the contents of a token
  - **IMPORT** - Adds a token certificate to RACF
- See appendix
- Access to the function and the token is controlled by permission checking in the CRYPTOZ class and/or the FACILITY class  
See “z/OS Security Server RACF Command Language Reference”, SA22-7687

R\_Datalib (IRRSDL00)

- Existing RACF (SAF) callable service to read key rings  
The service used by any application that needs to access X.509 certificates stored in RACF keyring
- Will now support new “\*TOKEN\*” user ID for reading certificates in tokens
  - Example, Keyfile directive in webserver’s httpd.conf file:

```
keyfile *TOKEN*/VENDOR.TOK SAF
```

PKCS#11 token label

Specifies that the application uses a PKCS#11 token instead of a keyring label

No change required here in the application to use certificates in tokens instead of RACF keyrings (e.g. IHS, System SSL, WAS, ..)

**PKCS#11 Tokens Management**

**System SSL gskkyman**

- gskkyman UNIX command line utility enhanced to manage PKCS#11 tokens similarly to key database (.kdb) files  
Requires ICSF up and running
  - Menu and command line driven
  - Create and delete tokens
  - List tokens
  - Manage a token's contents
- System SSL runtime can use tokens instead of key rings
  - Reads them indirectly from the TKDS through the SAF key ring services R\_Datalib using the \*TOKEN\*/<token-name> convention

**gskkyman main menu**

See « z/OS Cryptographic Services System Secure Sockets Layer Programming », SC24-5901

## Database Menu

```
1 - Create new database
2 - Open database
3 - Change database password
4 - Change database record length
5 - Delete database
6 - Create key parameter file
7 - Display certificate file (Binary or Base64 ASN.1 DER)

11 - Create new token
12 - Delete token
13 - Manage token
14 - Manage token from list of tokens

0 - Exit program

Enter option number:
===>
```

## Manage token

```
Token: HTTP.SERVER.TOKEN

Manufacturer: RACF HRF7740
Model: HCR7740
Flags: x00000509 (INITIALIZED,PROT AUTH PATH,USER PIN
INIT,RNG)

1 - Manage keys and certificates
2 - Manage certificates
3 - Manage certificate requests
4 - Create new certificate request
5 - Receive requested certificate or a renewal certificate
6 - Create a self-signed certificate
7 - Import a certificate
8 - Import a certificate and a private key
9 - Show the default key
10 - Delete Token

0 - Exit program

Enter option number (press ENTER to return to previous menu):
===>
```

# Thank You

# Any Questions ?



## Appendix

- z/OS V1R9 Cryptographic Services Manuals
  - *Writing PKCS #11 Applications (SA23-2231)*
  - *System Secure Sockets Layer Programming (SC24-5901)*
  - *ICSF System Programmer's Guide (SA22-7520)*
  - *ICSF Administrator's Guide (SA22-7521)*
  - *ICSF Application Programmer's Guide (SA22-7522)*
  
- z/OS V1R9 Security Server (RACF) Manuals
  - *Callable Services (SA22-7691)*
  - *Command Language Reference (SA22-7687)*
  - *Security Administrator's Guide (SA22-7683)*
  
- Information on PKCS #11
  - <http://www.rsa.com/rsalabs/node.asp?id=2133>

## New ICSF callable services

- CSFPTRC - Token Record Create
- CSFPTRD - Token Record Delete
- CSFPTRL - Token Record List
- CSFPSAV - Set Attribute Value
- CSFPGAV - Get Attribute ValueChanged information

## RACDCERT ADDTOKEN and DELTOKEN

- ADDTOKEN -
  - Syntax: **RACDCERT ADDTOKEN(token-name)**
  - Name must not already exist in ICSF
  - Tokens are not tracked by RACF after creation
  - e.g., `racdcert addtoken(http.server.token)`
- DELTOKEN
  - Syntax: **RACDCERT DELTOKEN(token-name) [FORCE]**
  - Destroys the token and all objects within it
  - FORCE keyword required if token contains objects not found in RACF
  - e.g., `racdcert deltoken(http.server.token)`

## Command line gskkyman

- Export certificate and associated private key
  - **gskkyman -e -t *token-name* -l *label* -p *file-name***
- Import certificate and associated private key
  - **gskkyman -i -t *token-name* -l *label* -p *file-name***
- Generate signed certificate
  - **gskkyman -g -x *days* -cr *file-name* -ct *file-name* -t *token-name* -l *label* -ca -ic**

### Options

- |                                    |                               |
|------------------------------------|-------------------------------|
| -ca Generate CA certificate        | -cr Certificate request file  |
| -ct Certificate file               | -l Label                      |
| -p Import/export file              | -t z/OS PKCS #11 token        |
| -x Number of days until expiration | -ic Include certificate chain |

## Installation and Setup

- Prerequisites
  - ICSF setup for secure key crypto
    - See ***ICSF System Programmer's Guide*** - Installation chapter
- Create the ICSF TKDS VSAM data set
  - Sample IDCAMS job in ***ICSF Writing PKCS #11 Applications***
- Modify ICSF Options Data Set
  - Add TKDSN entry for TKDS name (required)
  - Sysplex-wide consistency for CKDS and TKDS (optional)
    - Add SYSPLEXTKDS and SYSPLEXCKDS keywords

## Installation and Setup...

### Sample ICSF Options Data Set:

```

CKDSN ( CSF . CSFCKDS )
PKDSN ( CSF . CSFPKDS )
TKDSN ( CSF . CSFTKDS )
SYSPLEXCKDS ( YES , FAIL ( YES ) )
SYSPLEXTKDS ( YES , FAIL ( YES ) )
COMPAT ( NO )
SSM ( NO )
KEYAUTH ( NO )
CKTAUTH ( NO )
CHECKAUTH ( NO )
TRACEENTRY ( 1000 )
USERPARM ( USERPARM )
COMPENC ( DES )
REASONCODES ( ICSF )

```

## Installation and Setup...

- Setup PKCS #11 access control (Required)
  - Activate and RACLIST the CRYPTOZ SAF Class:
    - No PKCS #11 support without CRYPTOZ access
      - Message CSFM012I issued when ICSF is started
    - **SETR CLASSACT(CRYPTOZ) RACLIST(CRYPTOZ)**
  - Activate generics for the class if desired (recommended):
    - **SETR GENERIC(CRYPTOZ)**
  - Define USER and SO protection profiles and permit users as needed:
    - Create token naming convention - enforce with profiles
    - See samples in *ICSF Writing PKCS #11 Applications*
    - Refresh the CRYPTOZ Class whenever profiles change
      - **SETR RACLIST(CRYPTOZ) REFRESH**



## Installation and Setup...

- Modify ICSF callable service access control (Optional)
  - If customer controls access to CSFSERV class, then
    - New PKCS #11 callable services are also controlled
    - Define profiles for new services and permit users as needed, e.g.,  
**RDEF CSFSERV CSF1TRC UACC(NONE)  
 PERMIT CSF1TRC CLASS(CSFSERV) ACC(READ) ID(JUSER)**
  - Token management callable services are externalized
    - Documented in *ICSF Administrators Guide*
    - Access to these services is required for:
      - Token management via RACDCERT or gskkyman
      - ICSF's Token Browser ISPF panels
  - Operational callable services are internal only
    - PKCS #11 calling applications would still need access
  - See *ICSF Writing PKCS #11 Applications* for complete list

## Installation and Setup...

- (Re)start the ICSF started procedure
  - If configured properly, no new console messages displayed
    - Message CSFM012I indicates CRYPTOZ Class problems
      - Not activated or not RACLISTed
- Test your setup with testpkcs11
  - UNIX utility - /usr/lpp/pkcs11/bin/testpkcs11
    - Creates a temporary token and a key-pair then deletes
    - Source code shipped in /usr/lpp/pkcs11/samples