z/TPF V1.1

TPF Users Group - Spring 2009

Title: z/TPF Communications and Security Enhancements

Name: Mark Gambino
Venue: Communications Subcommittee

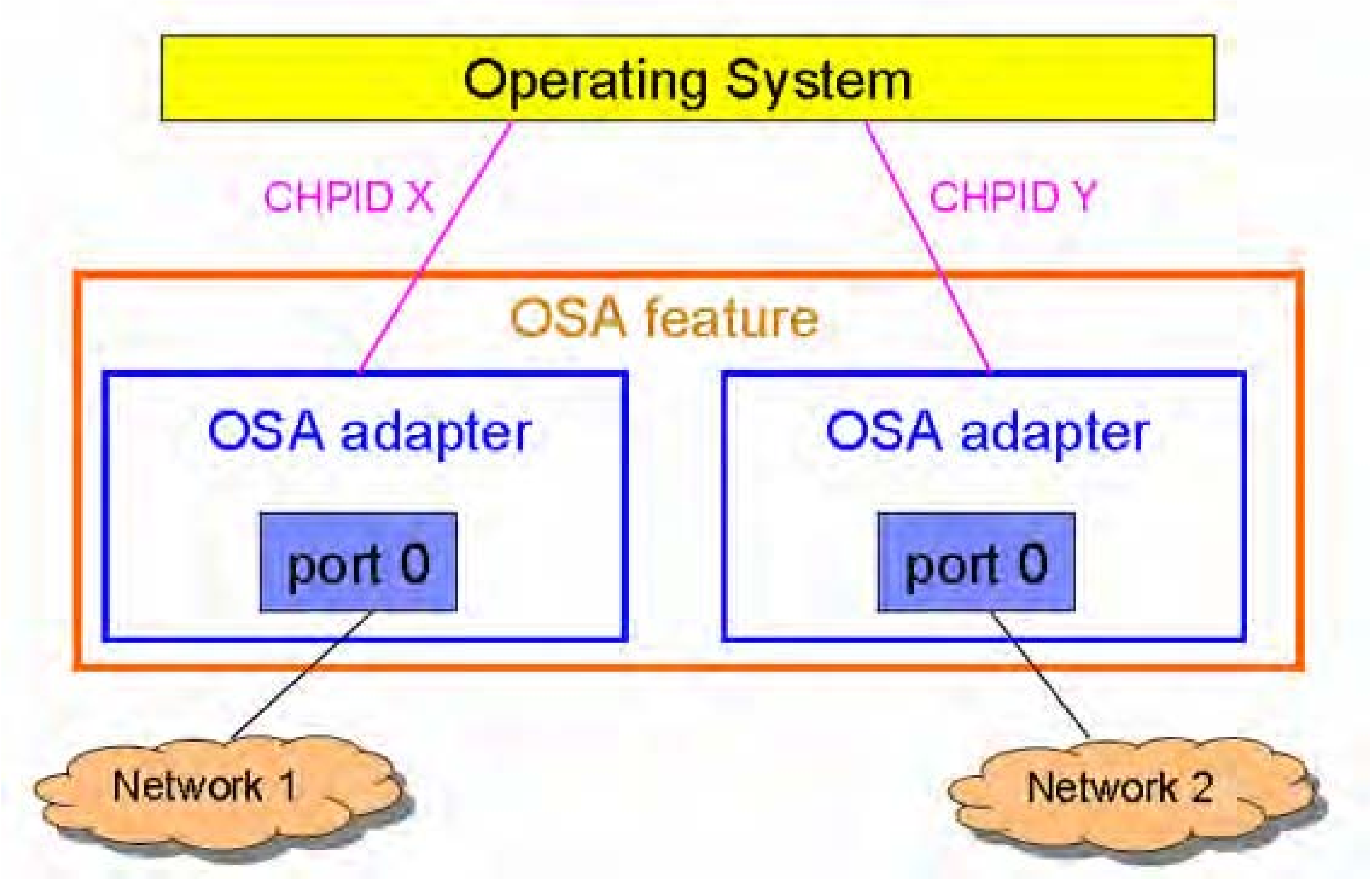
AIM Enterprise Platform Software
IBM z/Transaction Processing Facility Enterprise Edition 1.1.0

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.

OSA-Express2 and OSA-Express3 Terminology

- **OSA feature is the physical card that plugs into the I/O cage on the System z server**
- **OSA adapter is a physical OSA adapter residing on a physical card**
  - One OSA feature contains either 1 or 2 OSA adapters depending on type (like 1 GbE or 10 GbE)
  - Each OSA adapter is accessed via a separate CHPID
  - An OSA adapter can be shared by multiple LPARs
  - Prior to z10, each OSA adapter had a single network port connecting to an Ethernet network

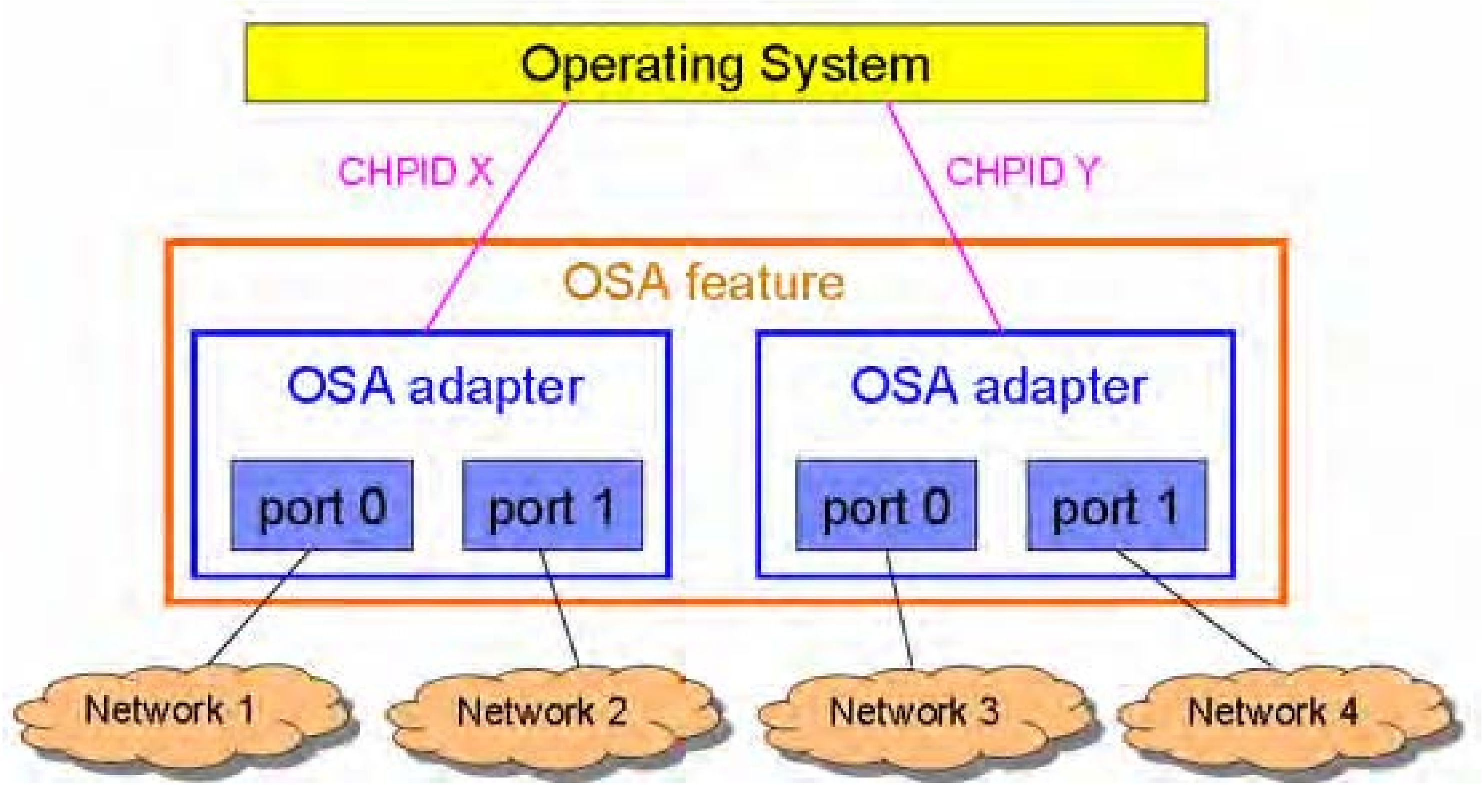
OSA Feature with 2 Adapters each with 1 Network Port



OSA Multiple Network Port Support

- **Some OSA-Express3 adapters on z10 support multiple network ports. For example:**
  - 1 GbE feature with 2 adapters each having 2 ports
  - 10 GbE feature with 1 adapter that has 2 ports
- **OSA adapter with a single network port has port 0**
- **OSA adapter with two network ports has port 0 and port 1**

OSA Feature with 2 Adapters each with 2 Network Ports



z/TPF OSA Multiple Network Port Support

- **APAR PJ32093 enables z/TPF to connect to different network ports on an OSA adapter**
  - Specify the PORTNUM parameter on the ZOSAE command that defines the OSA connection to z/TPF
  - Default is PORTNUM-0 (port 0)
- **All OSA connections that were defined to z/TPF prior to installing PJ32093 will use port 0**
- **z/TPF systems without PJ32093 applied can only use port 0 on all OSA adapters**

INETD Performance Enhancements

- **Enables Internet Daemon (INETD) listener ECBs to run on all I-streams to perform better in a tightly-coupled environment with multiple INETD servers**
  - Prior to this support all INETD listener ECBs ran on the same I-stream
- **INETD internal logic also changed to process connection requests and messages faster**
  - Applicable for network restart and high volume message rate conditions
- **These changes occur when you apply APAR PJ34780**
  - No INETD definition changes are needed

New NSD Message Weighting Option

- **Network Services Database (NSD) allows you to define properties of an application (where an application is defined by its TCP/IP port number)**
- **NSD resides in the *<i>/etc/services</i>* file**
- **The weight parameter in an NSD entry defines the weighting factor to use for input and output messages of this application when TCP/IP messages are reported in Data Collection**
- **APAR PJ35108 allows weight=0 to be specified**
  - weight=0 means count the messages for this application, but do not include them in the weighted message reports in data collection
  - You can display the raw message counts, even for applications coded as weight=0, using the ZIPDB command

Ability to Trace Variable Amount Data by Application

- **You can control how much data in each packet is traced by the system wide z/TPF IP trace facility**
  - Use the *ZTTCP TRACE SIZE-size* command
- **With APAR PJ35108 you can now specify how much data to trace on a per application basis**
  - Specify the IPTRSIZE parameter on the NSD entry of the application
  - If you do not specify IPTRSIZE for an application, the system wide value (from ZTTCP TRACE SIZE) is used

z/TPF Public Key Infrastructure (PKI) Support

APAR PJ32686

Public Key Cryptography

- **Also known as asymmetric key cryptography**
- **Public key pair**
  - Public key
    - Can be openly distributed
  - Private key
    - Must be kept secret/protected
- **Any data encrypted with a public key can only be decrypted using the corresponding private key**
- **Any data encrypted with a private key can only be decrypted using the corresponding public key**
- **Commonly used algorithm is Rivest, Shamir, and Adleman (RSA)**

Public Key Infrastructure (PKI)

- **Enables users of an unsecure (public) network to securely and privately exchange data**
- **Uses public key cryptography**
- **Public keys are shared/distributed using digital certificates that are created/signed by a trusted authority called a certificate authority (CA)**

z/TPF PKI Support Concepts

- **Allows you to create, manage, and use RSA key pairs in a secure manner**
  - Extends z/TPF secure key management support
- **Supports 1024-bit and 2048-bit RSA key pairs**
- **Key pair is referenced by name**
- **Private key value is secured – not visible to operators, applications, coverage, and so on**
- **Public key value is available to anyone**

ZPUBK GENERATE Command

- **Generates an RSA key pair and adds it to the PKI keystore**
- **Specify the name of the key pair (for example, KEYPAIR1)**
  - The key pair name is how subsequent operator commands and APIs will reference/use this key pair
- **Specify the key length (1024-bit or 2048-bit)**

```
ZPUBK GENERATE KEYPAIR-KEYPAIR1 CIPHER-RSA1024
```

```
PUBK0001I  KEYPAIR-KEYNAME1 BEING GENERATED
PUBK0002I  KEYPAIR-KEYNAME1 GENERATED AND ADDED TO PKI MASTER KEYSTORE
PUBK0003I  KEYPAIR-KEYNAME1 ADDED TO PKI MEMORY KEYSTORE ON ALL PROCESSORS
```

ZPUBK ACTIVATE Command

- **Activates an RSA key pair**

```
ZPUBK ACTIVATE KEYPAIR-KEYPAIR1

PUBK0011I  KEYPAIR-KEYNAME1 NOW ACTIVE IN PKI MASTER KEYSTORE
PUBK0012I  KEYPAIR-KEYNAME1 NOW ACTIVE IN PKI MEMORY KEYSTORE
           ON ALL PROCESSORS
```

Other Operator Commands

- **ZPUBK DEACTIVATE**
  - Deactivates an RSA key pair
- **ZPUBK DELETE**
  - Deletes an RSA key pair from the PKI keystore
- **ZPUBK DISPLAY**
  - Displays information about public key pairs in the PKI keystore
- **ZPUBK EXTRACT**
  - Extracts a public key to a file that can then be sent to a remote platform

Installation Instructions

1. **Define the PKI master keystore**
   - Define #IPKI fixed file records
2. **Define the PKI memory keystore**
   - Use the PKEYS macro in SIP
3. **Initialize the PKI master keystore**
   - Use the ZPUBK INITIALIZE and CONFIRM INITIALIZE commands
4. **Create RSA key pair(s)**
   - Use the ZPUBK GENERATE command
   - Must make a backup copy of the PKI keystore before continuing
     - Use the ZKEYS BACKUP command
5. **Activate RSA key pair(s)**
   - Use the ZPUBK ACTIVATE command

RSA Key Pairs are Created… Now What?

- **A z/TPF generated RSA key pair can be used:**
  - By Secure Sockets Layer (SSL) applications
    - Requires a digital certificate to be created that contains the RSA public key created by z/TPF
  - To securely import a symmetric key
    - You must send one of the following to the remote key manager that is exporting the symmetric key:
      - A digital certificate containing the z/TPF RSA public key
      - A file containing the z/TPF RSA public key (create the file using the ZPUBK EXTRACT command)

Creating a Digital Certificate

1. **z/TPF operator creates an RSA key pair called KEYPAIR1**
2. **Create a file (called myinfo.cfg in this example) containing the subject information needed to create a certificate request**
3. **Issue the ZPUBK REQCERT command. Input includes:**
   - Key pair name (KEYPAIR1) that says which public key to use
   - Name of the file (myinfo.cfg) containing the subject information
   - Name of the file (mycert.fil) into which to build the certificate request (PKCS #10 format)
4. **Send (FTP) the certificate request to the certificate authority (CA) that will create the digital certificate**
5. **From the CA, send (FTP) the certificate to your z/TPF system**

Self-Signed Certificates

- **A self-signed certificate is where the subject and issuer of the certificate are the same**
- **In production typically the only nodes that have/create self-signed certificates are CAs**
- **It is often convenient to have self-signed certificates for test systems**
- **To create a self-signed certificate on z/TPF**
  - Specify the SSIGNED option on the ZPUBK REQCERT command
  - Creates a certificate (self-signed) rather than a certificate request

OpenSSL Programming Model

- **z/TPF supports OpenSSL APIs**
- **SSL server applications must issue APIs specifying:**
  - Name of file containing the node’s digital certificate
  - Name of file containing node’s private key
- **When SSL client authentication is being used, SSL client applications must also issue those APIs**

Updating SSL Applications to Use z/TPF RSA Key Pairs

- **Application program still uses standard OpenSSL APIs to indicate the name of the file that contains the digital certificate**
- **Application program still uses standard OpenSSL APIs to indicate the name of the file that contains the private key**
  - Instead of pointing to a real file that contains the RSA private key, the application specifies a file name with a special prefix (/tpfpubk) that tells z/TPF that the name that follows is really the name of the RSA key pair to use and not to try and open/use a file in the file system

SSL Example of How to Specify a z/TPF RSA Private Key

- **SSL application wants to use key pair named KEYPAIR1**
- **Input to the *SSL_CTX_use_PrivateKey_file* API includes a pointer to the file name containing the private key**
  - Set the file name to **/tpfpubk/ keypair1.pem**
- **The private key value is *not* copied into the application program's memory space**
  - Only the key pair name (KEYPAIR1) is saved in the SSL structure in the application's memory space

Middleware Using OpenSSL Accessing RSA Private Keys

- **Some middleware opens/reads the private key file to determine whether the private key is encrypted and if so, prompts the operator for the password**
- **Some middleware also compares the public key information in the private key file to the public key in the certificate to make sure they match**
- **When you create an RSA key pair on z/TPF**
  - A dummy private key file is created in the /tpfpubk directory
  - This file includes:
    - An unencrypted private key (no password needed)
      - **This is NOT the real private key value!**
    - The real public key value
- **Allows middleware to continue to work without any changes to the middleware**
  - Middleware does not have access to the real private key value

SSL Considerations

- **SSL applications on z/TPF can use:**
  - Remote generated RSA keys
    - Keys do not reside in the PKI keystore
    - Private key resides in a file in the file system
    - This was the only method supported before PJ32686
  - z/TPF generated RSA keys
    - Keys reside in the PKI keystore
    - This is the method added by PJ32686
    - There is no impact to CPU performance using z/TPF generated keys compared to using remote generated RSA keys

Data Encryption Outside the Scope of SSL

- **If you create a symmetric key on z/TPF and one z/TPF complex is the only one that needs to use that key**
  - Key distribution is not an issue
  - Example – z/TPF application encrypts data before writing it out to the z/TPF database and that application reads the data (and decrypts) it later on
- **If you want to exchange data with another platform and the data is encrypted at the application level using a symmetric key**
  - A key distribution mechanism is needed

Solving the Symmetric Key Distribution Problem

- **z/TPF now supports RSA key wrapping to securely import symmetric keys**
- **Remote key manager that creates the symmetric key (KEY1) will wrap (encrypt) KEY1 using z/TPF's public key**
- **The remote key manager sends the encrypted KEY1 value to z/TPF**
  - How this is done is not architected (not standardized)
- **New *tpf_secure_key_import* API on z/TPF unwraps (decrypts) the KEY1 value (using z/TPF's private key) and adds KEY1 to the symmetric key keystore**
  - The KEY1 value is never in the clear during the key exchange

How Does the Remote Key Manager get Public Key Value

- **Remote key manager needs z/TPF’s public key to be able to securely send a wrapped symmetric key to z/TPF**
- **Can extract the public key on z/TPF**
  - Use the ZPUBK EXTRACT command
  - Creates a public key file containing the public key
  - Can send (secure FTP) the public key file to the remote key manager
- **The other (and more commonly used) option is to send z/TPF’s certificate to the remote key manager**
  - Certificate contains z/TPF’s public key

Initial z/TPF PKI Support Summary (APAR PJ32686)

- Create and manage RSA public key pairs in a secure manner on z/TPF
  - Create, activate, deactivate, display, delete RSA public key pairs
  - Backup and restore the PKI keystore
- Use the RSA keys generated on z/TPF to create digital certificate requests as well as self-signed digital certificates
- Enable z/TPF SSL applications and middleware to use private keys generated by z/TPF
- Ability to extract a public key from the z/TPF keystore that can be distributed to remote partners such as key managers
- Ability to import a symmetric key in a secure manner using public key cryptography

z/TPF PKI Support Phase 2 (statement of direction)

- **New operator command to display contents of a digital certificate in human-readable format**
- **New operator command to convert a digital certificate from PEM to DER format**
  - Some keys managers only support DER format
- **Allow applications and middleware to use RSA directly:**
  - APIs to encrypt/decrypt user data using RSA public key cryptography
  - APIs to create and verify RSA digital signatures

z/TPF Security Book

- **Security is a very important issue for all customers**
- **All security related information for z/TPF is being incorporated into a single *z/TPF Security* book**
- **New information is also being added to this book**
- **The book is coming soon to a z/TPF Information Center near you**

z/TPF Security Book Contents (subject to change)

- **Security Concepts**
- **Symmetric Cryptography with Clear Key APIs**
- **Message Digest APIs**
- **Secure Key Management**
- **Internet Security**
- **Secure Sockets Layer**
- **Protecting Data in Flight**
- **Protecting Data at Rest**
- **Protecting Data in Use**
- **z/TPF Operator Security**
- **Database Security**
- **Program Security**
- **Memory Security**

Trademarks

- IBM and System z10 are trademarks of International Business Machines Corporation in the United States, other countries, or both.
- Linux is a trademark of Linus Torvalds 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.