# My Adventures with TCP/IP Port Security and RACF on z/OS

Joel Tilton
RACF Engineer
Mainframe Evangelist
April 2015
NY & Tampa Bay RACF Users Group

# Disclaimers

- All products, trademarks, and information mentioned are the property of the respective vendors.
- Mention of a product does not imply a recommendation.
- Always test new profiles on a non-production system.
- Only you can prevent IPLs…
- No ports were harmed in the making of this presentation

# Agenda

- So what is a port anyway?
- Why Port Security with RACF
- EZB.PORTACCESS Profile Syntax
- SAFNAME Design
- Port Reservation Syntax
- Planning & Implementation Strategy
- SERVAUTH Class Activation
- Unreserved Ports – TCP & UDP
- Auditing Port Access
- Required PTFs
- Additional Resources
- Summary

# What is a Port?

- An IP address is used to route the message to your computer. Once it arrives there, TCP uses the port number to know which program like ftp or email to hand it to
- From a SERVAUTH perspective…
  - Any mainframe program binding to
  - and/or listening on a TCPIP Port
  - SYS1.TCPIP.PROFILE

# Why Port Security with RACF?

**NATIVE TCPIP**

- Reservation by Stepname
  - Can be spoofed
  - Violations not well logged
- Unreserved ports not easily controlled
- Low Ports protected by RESERVELOWPORTS

**RACF**

- Reservation by SAFNAME
  - SAFNAME last qualifier of SERVAUTH portaccess profile
  - Cannot be spoofed
  - Successes or Violations logged to SMF (type 80)
- Unreserved ports easily controlled
- Low Ports protected by RESERVELOWPORTS

# EZB.PORTACCESS Profile Syntax

**EZB.PORTACCESS.*sysname*.*tcpname*.*safname***

| Qualifier | Description | Recommendation |
|---|---|---|
| sysname | Local SMF ID | • Use * unless need for per system segregation |
| tcpname | TCPIP started task jobname | • Use * unless multiple stacks |
| safname | Esoteric name coded in port reservation | • Can be generic<br>• Plan appropriately |

# SAFNAME Design

- Use known protocol name as SAFNAME
  - HTTP, HTTPS, LDAP, LDAPS
  - … if appropriate
- Use generics in profile, as appropriate
  - HTTP*, LDAP*
  - … if appropriate
- Relationship
  - 1 or more port reservations to RACF profile

# Port Reservation Syntax – Single

- Port Reservation Syntax – single port

```
PORT
;Port Protocol     Stepname      SAF    SAFName
80    TCP          *             SAF    HTTP ;webserver
389   TCP          *             SAF    LDAP
```

  - With SAFNAME, stepname only needed to distinguish between two different port listeners

```
636 TCP LDAPDIR BIND 192.168.0.8 SAF LDAPD   ; LDAPDIR
636 TCP LDAPPKI BIND 192.168.0.9 SAF LDAPPKI ; LDAPPKI
```

# Port Reservation Syntax – Range

- Reserve Port Ranges

```
PORTRANGE
;Portrange  Length Protocol Stepname  SAF    SAFName
1000        51     UDP          *     SAF    OMEGAMON
```

  - Reserves 1000 through 1050 for Omegamon

- Use Only One
  - Reserve Individual Port
  - Reserve Port Range

# Single Port Overrides Range

- Reserve Same Port Collisions
  - Different SAFNAME
  - Only Single Port Reservation Used
- ICH408I
  - Call Security!
  - Update SYS1.TCPIP.PROFILE
  - IPL

# Example: TN3270 – RACF Profile

**`EZB.PORTACCESS.*.*.TN3270`**

- UACC always NONE
- Permit TN3270 STC user ID with READ
- AUDIT
  - ALL(READ)
    - Audit all port access attempts; failures and successes
    - Exclude FTP data port
- WARNING
  - Use sparingly as an implementation strategy

# Example: TN3270 – Reservation

```
PORT 23   TCP TN3270
```

- Non-SAF uses stepname

```
PORT 23   TCP *          SAF TN3270
```

- With SAF
  - Stepname unnecessary
  - Only use stepname where needed

# Planning – Gather Information

- Evaluate running STCs and their ports
  - NETSTAT CONN ➔ What is Listening
  - NETSTAT PORTLIST ➔ How it is Reserved
    - REMINDER: SERVAUTH EZB.NETSTAT.**
  - REXX EXEC compare reservations vs. usage
- Create list of Port Listeners & SAF*names*
- Partner with Network/VTAM Engineer
  - TCPIP profile changes
  - Weekend IPLs
- Update Software Parms
- Implement one system at a time
  - development, test and then production

# Planning – Implementation

- Build EZB.PORTACCESS profiles
  - No effect until SYS1.TCPIP.PROFILE Updated
  - TCP Ports – OMPROUTE READ 
- Update port reservations to call SAF
- Activate via IPL or TCPIP OBEY
  - Large number of STCs ➔ IPL
  - OBEY command is dynamic
    - Cycle Started Tasks
    - Excludes FTP

# Planning – Intermittent Listeners

- NETSTAT CONN
  - Shows ports in use *now*
  - Not every port is in constant use by its listener
- Find *intermittent* port listeners
  - WARNING AUDIT(ALL(READ))
    - EZB.PORTACCESS.*.*.UNRSVTCP
  - Mine SMF records
  - Midnight Logons – Optional
- Update TCPIP profile Port Definitions
  - RDEFINE SERVAUTH EZB.PORTACCESS.*.*.SAFname
  - PERMIT EZB.PORTACCESS.*.*.SAFName class(SERVAUTH) access(READ) ID(STC UserID)
- IPL
- Rinse, Recycle, Repeat …

# SERVAUTH Class Activation

- Activate SERVAUTH Class
  - IBM Class Descriptor Table (CDT)
  - SETR classact(SERVAUTH) audit(SERVAUTH) raclist(SERVAUTH) generic(SERVAUTH)
    - RC of 4 class but be mindful of SYS1.TCPIP.PROFILE
      - SERVAUTH profiles for DVIPA
      - EZD1313I -REQUIRED SAF SERVAUTH PROFILE NOT FOUND *RACF profile name*
- RDEFINE RACGLIST SERVAUTH
  - Performance Improvement
  - SETR classact(RACGLIST) audit(RACGLIST)
  - SETR RACLIST(…) REFRESH Creates

# Required PTFs – Apply FIRST

- Spurious SAF (RACF) Violations from use of UDP Sockets
  - APAR PI18153 PTF UI8700 for z/OS 1.13
  - APAR PI18151 PTF UI9430 for z/OS 2.1
  - Use of UDP Ephemeral ports causes random security violations
  - http://www-01.ibm.com/support/docview.wss?uid=isg1PI18151
- ABEND S0C4 IN EZBXFUT6
  - APAR PI07541 PTF UI13629 for z/OS 1.13
  - APAR PI08351 PTF UI14006 for z/OS 2.1
  - PORT UNRSV TCP * SAF SAF*name*
  - Mapping via SRCIP to a DVIPA with sysplexports defined
  - Does not have permission to the SAF resource
  - http://www-01.ibm.com/support/docview.wss?uid=isg1PI08351

# Unreserved Ports Syntax

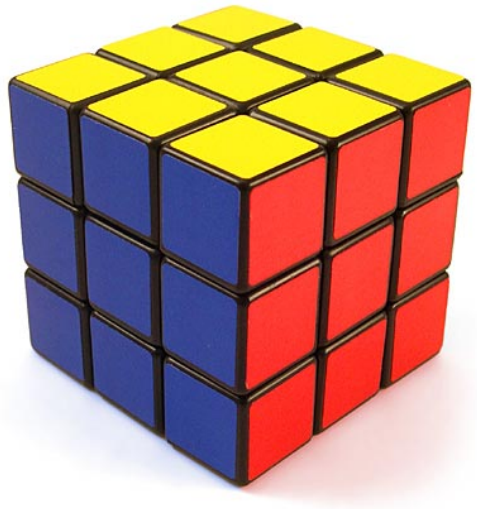
- `PORT UNRSV TCP * SAF` **UNRSVTCP**
  - Prevent TCP port listeners → **TCP default**
- `PORT UNRSV TCP * SAF` **UNRSVTCP** WHENBIND
  - Prevent TCP client port binds → **optional**
- `PORT UNRSV UDP * SAF` **UNRSVUDP**
  - Prevent UDP port listeners & binds → **UDP default**
- Stop Unauthorized Port Use
  - Goal: Emtpy ACLs
  - AUDIT(ALL(READ)) UACC(NONE)
- Consideration: Dynamic Ephemeral UDP ports
  - See Required PTFs

# Unreserved Ports Profile Syntax

- Build SERVAUTH Profiles
  - EZB.PORTACCESS.*.*.**UNRSVTCP** OWNER(…) UACC(NONE) WARNING AUDIT(ALL(READ))
    - Permit all STCs in the Beginning
  - EZB.PORTACCESS.*.*.**UNRSVUDP** OWNER(…) UACC(NONE) WARNING AUDIT(ALL(READ))
  - Read SMF, Read SMF, Read SMF…Logstring
  - Update Product Parms
  - Cautiously Restrict Access

# UDP Unreserved Ports – SOLVED

- UDP Ephemeral ports
  - Applications Need Dynamic Ephemeral UDP
  - STC desires to use SMTP to send email
  - STC opens UDP Ephemeral port on demand
    - SMTP
  - Triggers SAF call unless…
  - PTF UI8700 for z/OS 1.13
  - PTF UI9430 for z/OS 2.1



# TCP Unreserved Ports – Omegamon

- Binds ports to loopback (127.0.0.1)
  - New PTF under development
- Multiples of 4096 + a base number

- 1918 + (n * 4096) = Agent Port Number
  - Where N = The startup agent number.
- 1918 – Base Port, always assigned to hub or remote TEMS (Omegcms)
- 1920 – IBM Tivoli Monitoring Service Console (assigned to first agent to start up, OMEG*)
- 6014 – MVS Agent
- 10110 – CICS Agent
- 14206 – Network Agent

# TCP Unreserved Ports – Omegamon

```
  6014 TCP * SAF OMEGAMON ; OMEGAMON
 10110 TCP * SAF OMEGAMON ; OMEGAMON
 14206 TCP * SAF OMEGAMON ; OMEGAMON
 18302 TCP * SAF OMEGAMON ; OMEGAMON
 22398 TCP * SAF OMEGAMON ; OMEGAMON
 26494 TCP * SAF OMEGAMON ; OMEGAMON
 30590 TCP * SAF OMEGAMON ; OMEGAMON
 34686 TCP * SAF OMEGAMON ; OMEGAMON
 38782 TCP * SAF OMEGAMON ; OMEGAMON
 42878 TCP * SAF OMEGAMON ; OMEGAMON
 46974 TCP * SAF OMEGAMON ; OMEGAMON
 51070 TCP * SAF OMEGAMON ; OMEGAMON
 55166 TCP * SAF OMEGAMON ; OMEGAMON
 59262 TCP * SAF OMEGAMON ; OMEGAMON
 63358 TCP * SAF OMEGAMON ; OMEGAMON
PORTRANGE  1900  51 TCP *        SAF OMEGAMON
PORTRANGE  1900  51 UDP *        SAF OMEGAMON
```

# TCP Unreserved Ports – Challenges

- WAS Admin console scans ports
  - JVM Setting available for WAS v7.0, v8.0 and v8.5
  - PTF & Still Investigating
- z/OS FTP Client
  - If Passive FTP fails, attempt Active
  - Active connection Listens on TCP port
  - No Parm to Disable Active Mode

# WAS APAR PM96838

- Available for WAS v7.0, v8.0 and v8.5
- PM96838 – Optionally disable port activity checking when a server is created
- **com.ibm.ws.management.suppressPortScan=true**
  - JVM argument is added to suppress port check
- Note that when this is in effect, ports in use by other applications will not be detected and could lead to port conflicts.

# PTFs Under Construction

- z/OS FTP Client
  - First Attempts Passive FTP
  - If Passive FTP Failure, Active FTP
    - Listens on Random Port
- Omegamon
  - Loopback Interface Ports
  - Bind Port Range

# TCP WHENBIND – Challenges

- SAS
  - E-mail engine uses ports
  - TCP_EPH_MAP_ENABLED=**0** → zero
  - TKMVSENV DD
    - hlq.TKMVSENV(TKMVSENV)
- CA MICS
  - E-mail engine uses ports
  - Still Investigating
- VPS
  - Remove parm → TCPHOSTS

# Implementation Strategy

- Apply required PTFs
- Activate SERVAUTH class (RACGLIST too!)
- Known TCP & UDP Ports – Phase 1
  - Profiles in WARNING as appropriate
- "Midnight Madness" Port Listeners – Phase 2
- Secure Unreserved UDP ports – Phase 3
- Secure Unreserved TCP ports – Phase 4
- Secure Unreserved TCP client binds – Phase 5
- Goal:  Empty ACLs for Unreserved ports

# Auditing Port Access

```
22Nov14 12:14:31.11 OMEGC ZOS1 RACF ACCESS success for OMEGC: (READ,READ)
on SERVAUTH EZB.PORTACCESS.sysname.TCPIP.OMEGAMON

Jobname + id: OMEGCMS  STC12345

Class       : SERVAUTH Resource: EZB.PORTACCESS.ZOS1.TCPIP.OMEGAMON

Access used : READ      Profile: EZB.PORTACCESS.*.*.OMEGAMON

Log string  : TCPIP PORT ACCESS CHECK PORT 01000
```

- RACF total control of all ports
- All Port usage recorded in SMF
- LOGSTRING contains the port number
  - TCP / UDP Not Specified

# Additional Resources

- Techdocs Library – Using SERVAUTH to Protect TCP Port Usage
  - http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP100673
- Techdocs – Undesired PortAccess Violations
  - http://www-01.ibm.com/support/docview.wss?rs=852&uid=swg21237916
- Port Access Control Chapter
  - z/OS Communications Server: IP Configuration Guide
  - http://www-01.ibm.com/support/knowledgecenter/SSLTBW_2.1.0/com.ibm.zos.v2r1.halz002/security_tcpip_resrcs_ports.htm
- SERVAUTH Class profiles used by TCP/IP
  - EZB.PORTACCESS syntax
  - http://www-01.ibm.com/support/knowledgecenter/SSLTBW_2.1.0/com.ibm.zos.v2r1.halz002/security_tcpip_resrcs_saf.htm

# Summary

- Try not. Do…or do not. There is no try!
  - Master Yoda
- How do you tackle any project? One small step at a time…
- Protecting Ports is of Paramount Im**PORT**ance
  - Securing with RACF
    - prevents spoofing
    - logs port usage (success & failures) to SMF
- Requires Proper Planning
- Close partnership with Network Engineer
- Coordinate TCPIP Profile & RACF Changes
- IPL during maintenance windows
- Fix ICH408Is and:
  - Recycle STC or possibly IPL
- Port Security Engaged!

# My Thanks To…

- Adam Klinger
- Hayim Sokolsky
- IBM Omegamon Level 2 & Level 3
- IBM z/OS Comm Server/TCPIP Level 2 & Level 3
- IBM zSecure Level 2 & Level 3
- IBM WAS Level 2 & Level 3

- And the Adventure continues …

- **DISCLAIMER:** No ports were harmed in the making of this presentation

# Questions?