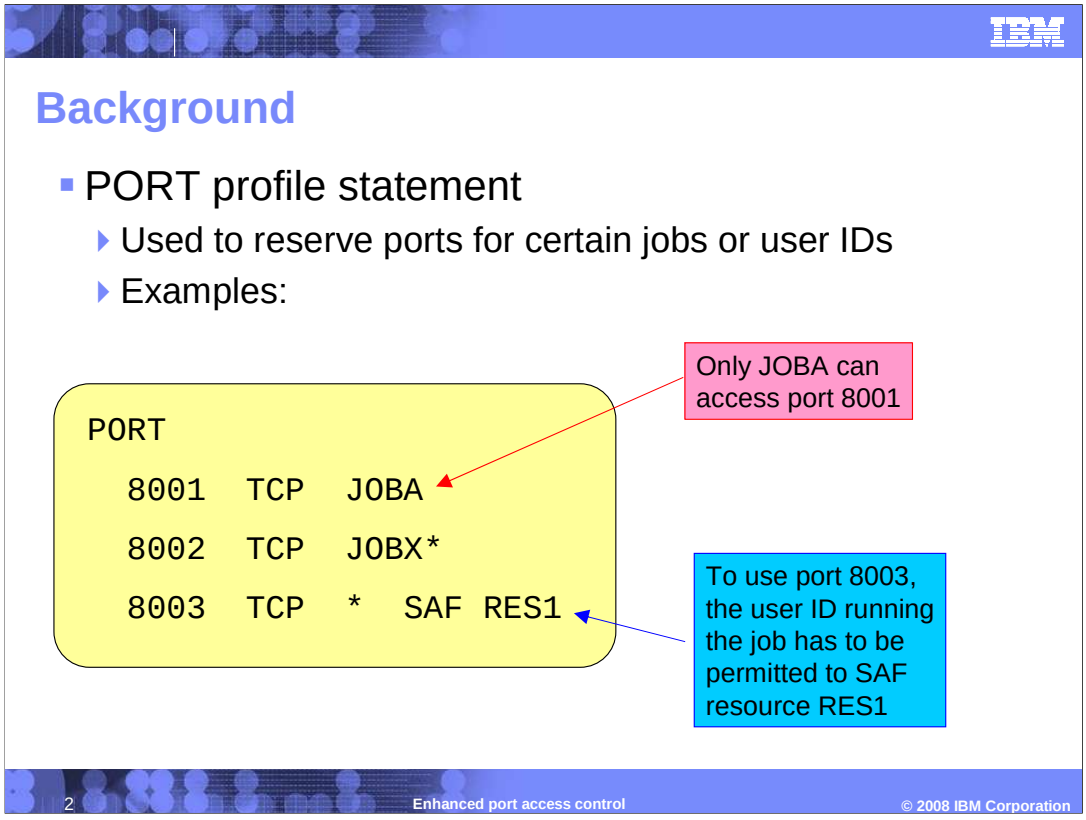
This presentation will cover enhanced port access control.



Ports can be reserved by configuring a PORT profile statement in your initial profile data set or in an OBEYFILE data set. When you reserve a port number, use of that port is limited to applications running under the specified job name or under a user ID that is authorized by the System Authorization Facility. If there are no PORT entries for a specific port number, any application can use that port.

The example shown here restricts access to three ports: 8001, 8002, and 8003. Port 8001 can only be used by JOBA. If any other job tries to use port 8001 (by issuing a bind socket call that specifies port 8001) the bind fails. Port 8002 can be used (if it is not already in use) by any job whose job name starts with the letters “JOBX”. To use port 8003, the job name does not matter (the asterisk means “all job names” can bind to this port). However, the keyword ‘SAF’ indicates that the user ID running the job is to be validated by the System Authorization Facility. The user ID must be permitted to the SAF resource EZB.PORTACCESS.sysname.tcpname.RES1 or the bind fails.

Unreserved ports

- Ports that are *not* reserved by the PORT profile statement
  - Unreserved ports below 1024 can be restricted to APF-authorized programs or Super user (UID(0)) authority
    - RESTRICTLOWPORTS parameter on the TCPCONFIG or UDPCONFIG profile statement
  - Unreserved ports from 1024 through 65535
    - Used by the stack when an ephemeral port is requested
    - Available to any job or user ID

Port numbers that are not specified on a PORT profile statement are considered "unreserved ports".

You can restrict the use of unreserved ports below 1024 to programs that are APF-authorized or have OMVS super user authority. You do this by configuring the RESTRICTLOWPORTS parameter on the TCPCONFIG or UDPCONFIG profile statements. However, the unreserved port numbers from 1024 through 65535 are available for use by any application that issues an explicit bind to a specific unreserved port. These port numbers are also used by the stack to provide stack-selected ephemeral ports.

Controlling unreserved ports

In V1R9 and earlier releases, any application can

- Choose and bind to any unreserved port
  - If RESTRICTLOWPORTS, above port 1023
- Create a listening server on an unreserved port chosen by the application

Before V1R10, there is no way to prevent an application from binding to an application-specified unreserved port above port 1023. Only the port already being in use prevents this binding. There is also no way to prevent an application from creating a listening server that uses a unreserved port.

Controlling unreserved ports

- Several issues:
  - Some administrators want to be aware of all applications acting as TCP servers on the system
  - and control who can act as a server application
- There are be cases in which the chosen port is not available
  - it was allocated by TCP/IP as an ephemeral port
    - Leads to infrequent failures that are difficult to diagnose

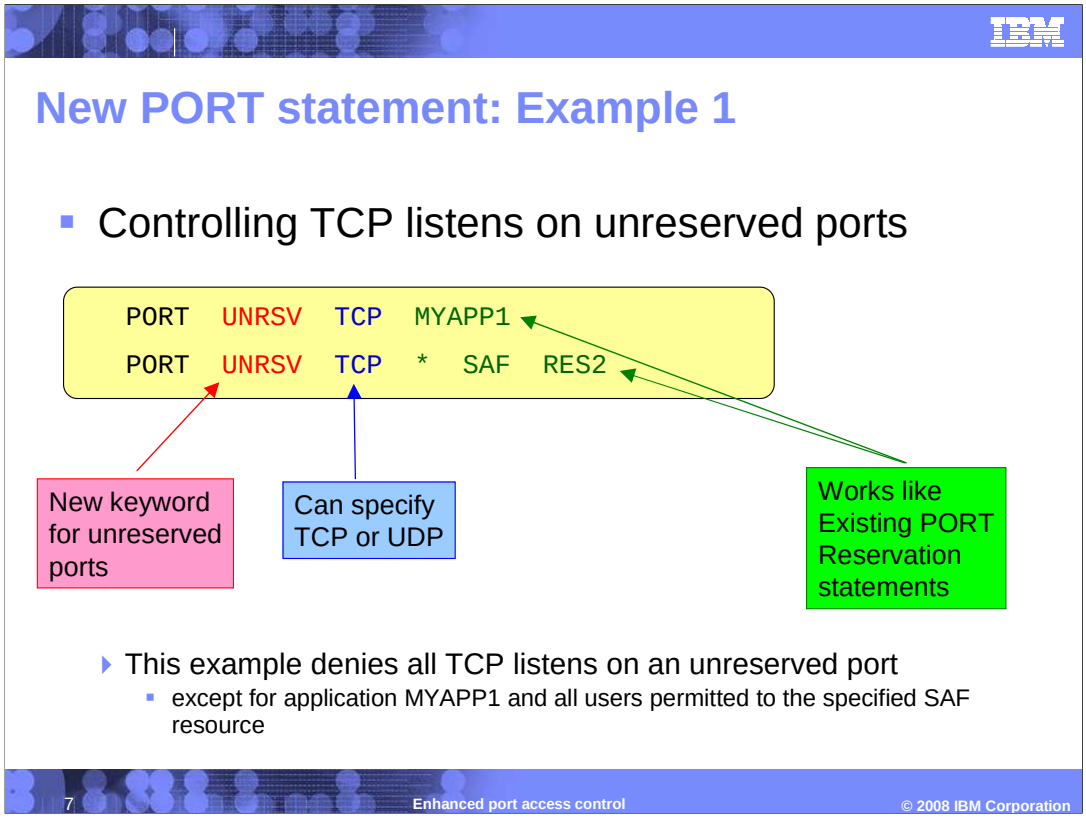
In some customer installations, administrators want to be aware of all applications acting as TCP servers. In addition, they want to control which applications can act as TCP servers and currently there is no way to control this.

2) When applications choose a particular unreserved port to use, there is no guarantee that the port is available. While the “chosen” port might be available most of the time, there might be cases where the unreserved port has already been allocated by TCP/IP as a ephemeral port. This can result in infrequent failures that are difficult to diagnose.

Enhanced port access control

- In V1R10, an expanded PORT profile statement provides a way to control application ability to
  - Choose a specific unreserved port
  - Listen on an application-chosen unreserved port
- Affects application-selected ports only
  - Stack-selected (for example, ephemeral, sysplex-wide, or for a DVIPA) ports not affected

V1R10 Communications Server expands the PORT profile statement to enhance control over application access to ports. Specifically, it provides a way to control which applications have the ability to choose an unreserved port by explicitly binding to it. For the TCP protocol, you can control which applications can establish a listening socket using an application-selected unreserved port.



The example on this slide shows the use of two PORT statement entries to control TCP listens on unreserved ports that were specified by the application on an explicit bind.

The first entry unconditionally allows TCP listen access for the application running under the job name MYAPP1.

The second entry allows TCP listen access for any job name whose user ID is permitted to the SAF resource EZB.PORTACCESS.sysname.tcpname.RES2.

As with port reservation entries, if there are multiple entries for a protocol, the entry with the closest match to the application's job name is used. So when the job name is MYAPP1, the first entry is used and there is no restriction on the user ID. For any job name other than MYAPP1, the second entry is used and the user ID must pass the SAF authorization check. If it does not, a TCP listen on an application-specified unreserved port fails.

New PORT statement: Example 2

- Controlling TCP binds on unreserved ports

```
PORT   UNRSV   TCP   WHENBIND MYAPP1
PORT   UNRSV   TCP   WHENBIND *   SAF   RES2
```

The default value is WHENLISTEN, so it was not specified on the previous slide

- Restriction: cannot mix WHENBIND and WHENLISTEN for TCP entries
- This example denies all TCP binds on an unreserved port
  - except for application MYAPP1 and all users permitted to the specified SAF resource

The example on this slide shows the use of two PORT statement entries to control TCP binds on unreserved ports that were specified by the application on an explicit bind. Remember that WHENLISTEN is the default access control for the TCP protocol.

The first entry unconditionally allows TCP bind access for the application running under the job name MYAPP1.

The second entry allows TCP bind access for any job name where the user ID is permitted to the SAF resource EZB.PORTACCESS.sysname.tcpname.RES2.

As with port reservation entries, if there are multiple entries for a protocol, the entry with the closest match to the application's job name is used. So when the job name is MYAPP1, the first entry is used and there is no restriction on the user ID. For any job name other than MYAPP1, the second entry is used and the user ID must pass the SAF authorization check. If it does not, a TCP bind on an application-specified unreserved port fails.

New PORT statement: Example 3

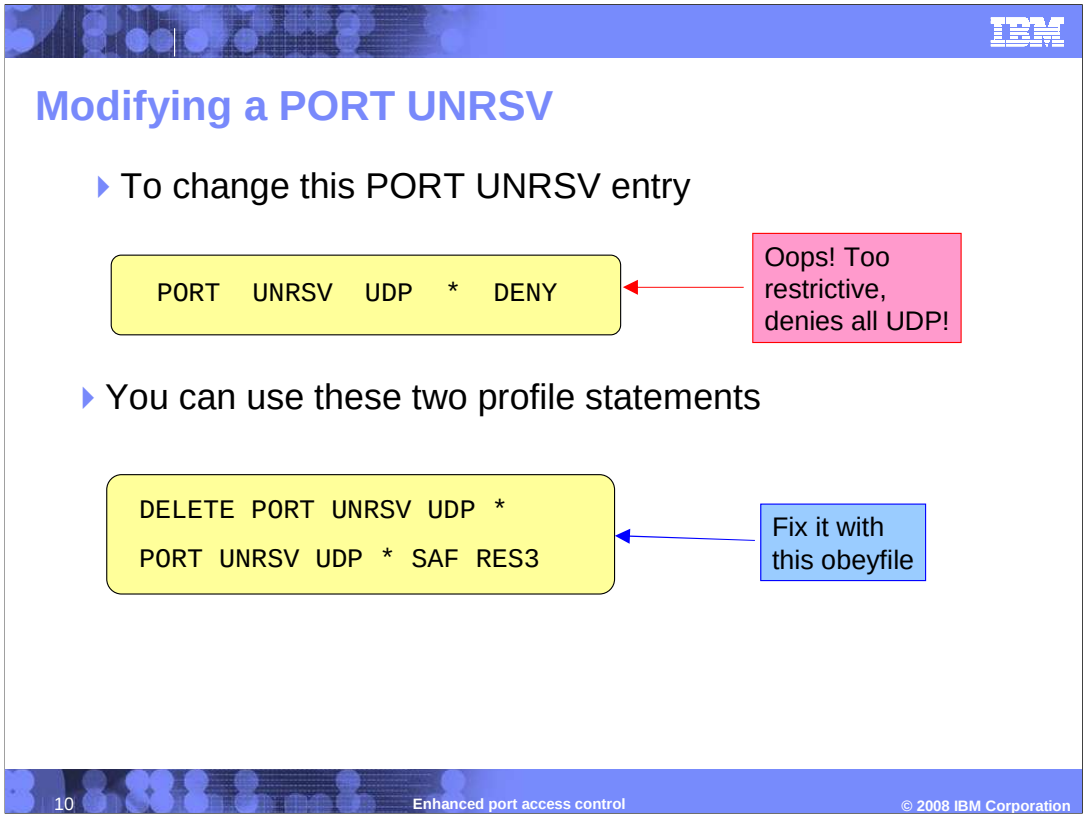
- Controlling UDP explicit binds to unreserved ports

```
PORT   UNRSV   UDP   *   SAF  RES2
```

  - denies all UDP explicit binds to an unreserved port
    - except for users permitted to your SAF SERVAUTH resource
  - No WHENLISTEN/WHENBIND keyword for UDP
    - WHENBIND is only method supported

The example on this slide shows a PORT statement entry that allows a UDP explicit bind to a non-zero unreserved port only for user IDs permitted to SAF resource EZB.PORTACCESS.sysname.tcpname.RES2.

All other UDP explicit binds to non-zero unreserved ports are denied.



This slide first shows a PORT statement that denies UDP explicit bind access to application-specified unreserved ports by any job.

It then shows two profile statements: a DELETE PORT statement that deletes the existing PORT UNRSV statement, followed by a new PORT UNRSV statement that replaces the original statement. Note that the keyword DENY is not required on the DELETE PORT statement.

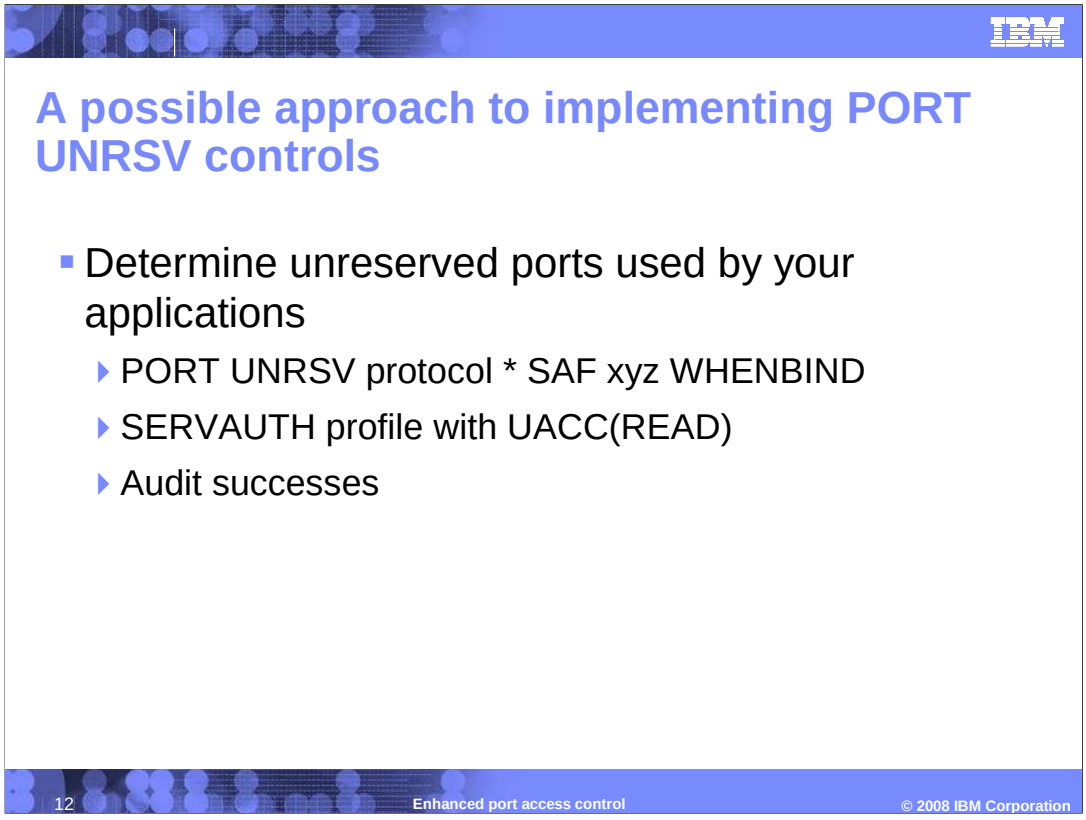
The new PORT UNRSV statement shown in this slide will allow UDP explicit bind access to application-specified unreserved ports by any job whose user ID is permitted to the specified SAF resource. In this case the resource is EZB.PORTACCESS.sysname.tcpname.RES3.

The DELETE PORT statement and new PORT UNRSV statement can be contained in the same OBEYFILE or in two separate OBEYFILEs.

Important caution on unreserved ports

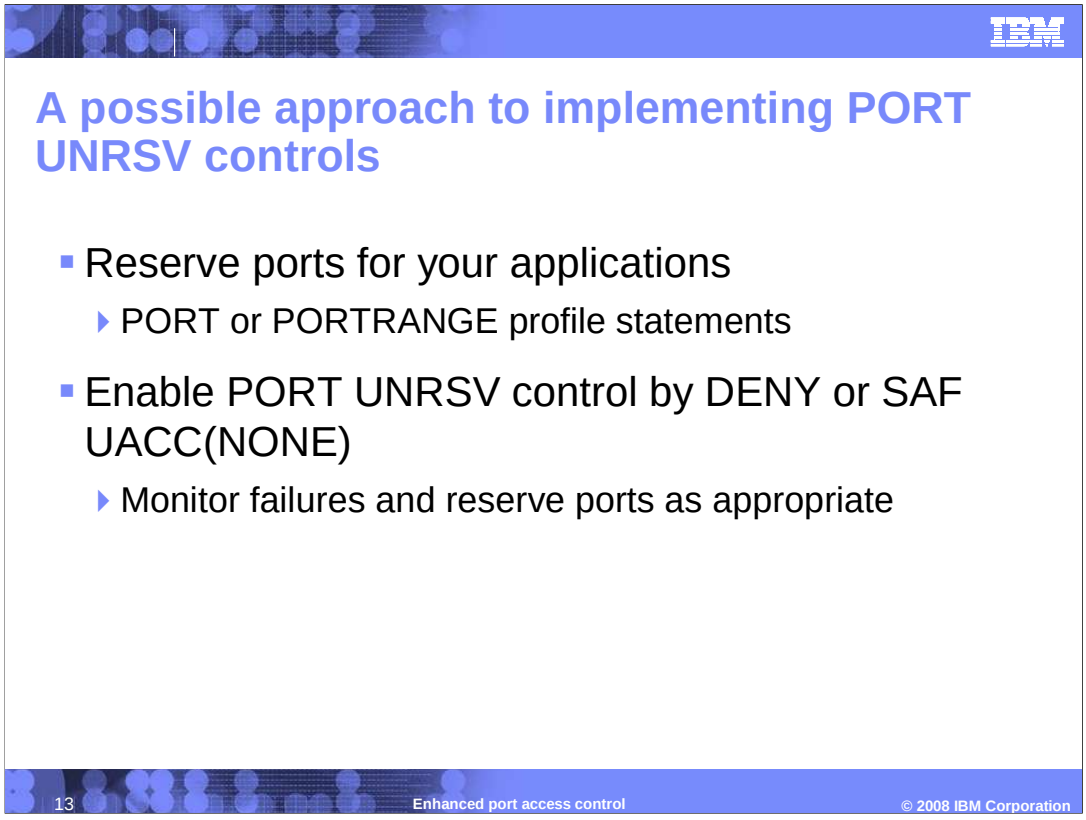
- PORT UNRSV controls can have broad and unexpected consequences
  - For example: Client programs can run under many different user IDs
    - so all address spaces where the client program can run need to be authorized.

There are no migration concerns with this function. However, using PORT UNRSV controls can have broad and unexpected consequences especially for client programs that run under different job names or user IDs . For example, if WHENBIND is used, any client program that explicitly binds to a non-zero unreserved port must be authorized for all address spaces where it can run.



Because of the potential consequences of using this function, you should consider these steps:

First, gather information about the ports used by your applications. For example, you might start out with configuring 'PORT UNRSV TCP * SAF xyz WHENBIND' where the SERVAUTH profile has UACC(READ) and auditing of successes is turned on. This will give you an indication of how many TCP applications currently bind to non-reserved ports and what ports they bind to.



Second, verify the applications reported and if deemed valid, reserve their ports using the PORT or PORTRANGE profile statements.

Third, enable access controls. When you are confident that you have configured the necessary port reservations statements for your applications, enable unreserved port access control. Do this by changing the PORT UNRSV statement to specify DENY, or by changing the SAF profile to UACC(NONE) and monitoring failures. As failures are identified, configure appropriate PORT reservations statements to allow authorized application access to those ports.

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM z/OS

A current list of other IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance 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 or performance improvements equivalent to the ratios stated here.