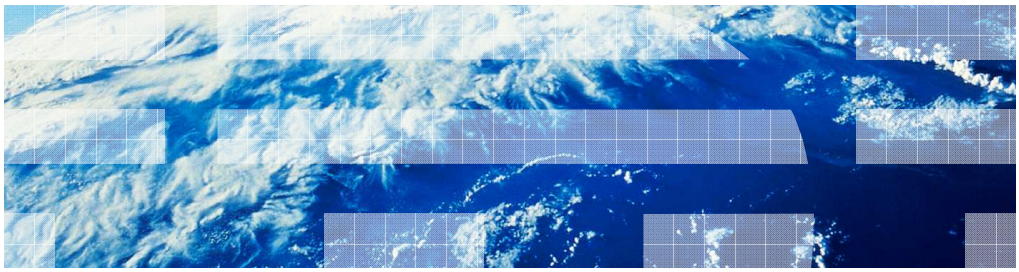
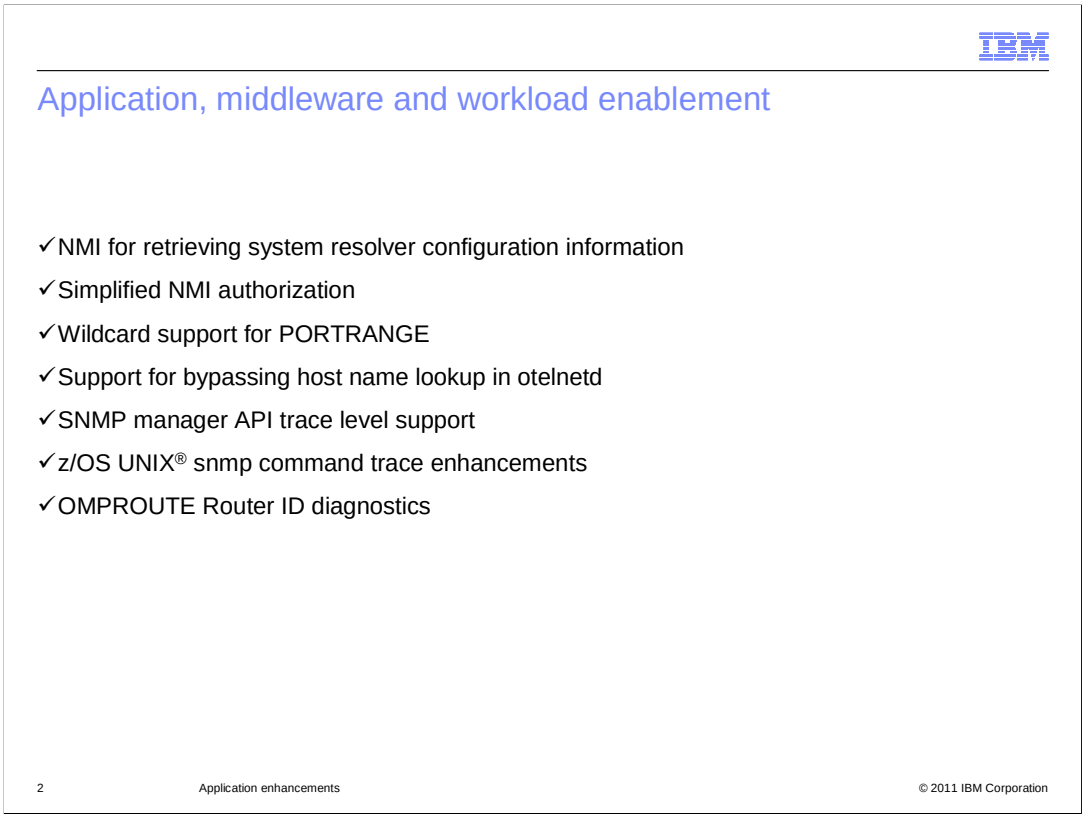
z/OS Communications Server –
Application enhancements



This presentation describes the updates to several applications in z/OS® V1R13 Communications Server for the application, middleware, and workload enablement theme.



The application, middleware, and workload enablement theme includes enhancements to network management interfaces, otelnetd, SNMP, and OMPROUTE.

NMI for retrieving system resolver configuration information

- New EZBREIFR callable network management interface (NMI)
  - Supports GetResolverConfig request
  - Retrieves global resolver configuration settings
  - Request and response data structured similarly to TCP/IP callable NMI (EZBNMIFR)
    - Calling application provides output buffer to hold the response information
    - Calling application must be authorized
    - Supports C/C++ and Assembler
    - Supports AMODE(31) and AMODE(64)

A new resolver callable NMI provides a fast interface for network applications to access resolver configuration data. A single request (GetResolverConfig) is supported. The resolver NMI does not provide any filtering options to reduce the amount of response data generated.

The resolver callable NMI is modeled after the TCP/IP callable NMI. You use the same triplet and quadruplet structures to identify the offset, length and number of various types of information on requests and responses. The calling application, which must be authorized, provides an output buffer area to hold the NMI response data.

NMI for retrieving system resolver configuration information: data provided

- Resolver configuration data provided:
  - Resolver setup statement values
  - Global TCPIP.DATA statement values

The resolver NMI provides the current resolver setup statement and global TCPIP.DATA statement values. The resolver indicates in the NMI output whether the configuration statements were explicitly defined or were defaulted. The resolver also includes the source file names of the MVS or file system files from which the configuration data was retrieved.

Locally defined TCPIP.DATA statements can override a subset of global values if the statements are not explicitly defined in the global file. These locally defined values are not returned in the NMI response. The resolver only provides the global defaults in the NMI response data.

NMI for system resolver configuration information: function externals

- Data mappings
  - EZBRENMA assembler macro
  - EZBRENMC C/C++ header
- *z/OS V1R13 Communication Server IP Programmer's Guide and Reference*
  - NMI field descriptions
  - Return and reason code descriptions

z/OS V1R13 Communications Server provides both assembler and C data mappings for the resolver NMI data. The *z/OS V1R13 Communication Server IP Programmer's Guide and Reference* includes descriptions of the data fields returned by the GetResolverConfig request and the possible return and reason codes returned when the request fails.

Simplified authorization requirements for real-time TCP/IP NMI (1 of 2)

- Real-time TCP/IP network monitoring NMI
  - SYSTCPDA: real-time TCP/IP packet trace data NMI
  - SYSTCPCN: real-time TCP connection SMF data NMI
  - SYSTCPOT: real-time OSAENTA packet trace data NMI
  - SYSTCPSM: real-time SMF data NMI

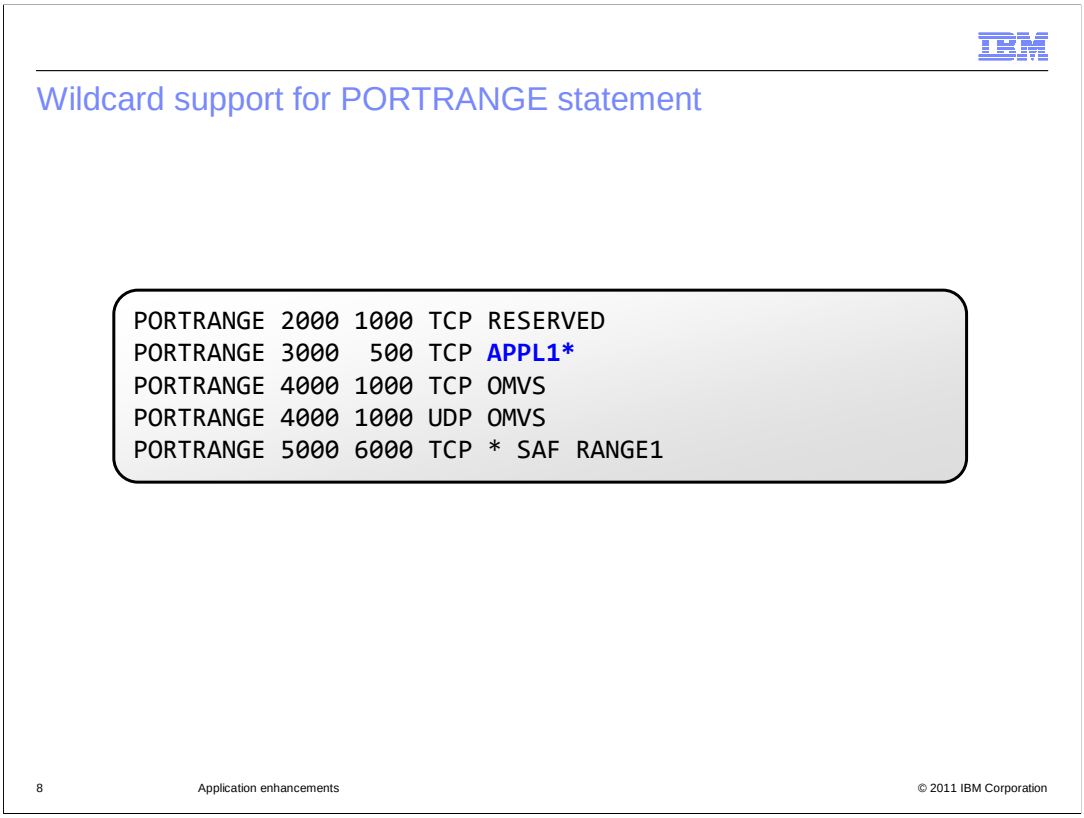
One of the network management interfaces (NMIs) provided by z/OS Communications Server allows for real-time network monitoring. There are several variations of this interface, providing real-time data about packet trace, TCP connections, OSAENTA packet trace and TCP/IP related SMF records.

Simplified authorization requirements for real-time TCP/IP NMI (2 of 2)

- Authorization requirements
  - Connection: one of
    - Access to EZB.NETMGMT.*sysname.tcpprocname.interface* in SERVAUTH class
    - Superuser
    - Access to BPX.SUPERUSER in FACILITY class
  - Copy-buffer: APF authorization
- New requirements for copy-buffer authorization
  - APF authorized, or
  - READ access to EZB.NETMGMT.*sysname.tcpprocname.interface* in SERVAUTH class and user ID unchanged

These interfaces have two separate authorization requirements. First, in order to connect to the interface, three alternate means of authorization are provided, as listed on this slide. Second, in order to copy data records from the interface after connecting, APF authorization is required.

Beginning in V1R13, z/OS Communications Server simplifies the authorization requirements for the real-time NMI. The copy-buffer authorization requirements are changed to allow either APF authorization, or READ access to the same profile in the SERVAUTH class that is used to restrict connection access. Applications or installations for which APF authorization is not allowed can now use the SAF security profile method of authorization to allow the network management application to copy data records. If this method of access is used, it is important that the application operates under the same user ID for both connection and copy-buffer operations.



z/OS V1R13 Communications Server introduces support for partial wildcarding of the job name on the PORTRANGE TCP/IP profile statement. This support is already available on the PORT statement. Applications whose job name prefix matches the given prefix are allowed to bind to ports within the port range. Wildcarding is supported for both TCP and UDP port ranges.

Wild carded job names are displayed in NETSTAT PORTLIST output.

Support for bypassing host name lookup in otelnetd

- New parameter, –g, disables host-name lookup of client IP address
  - Avoid messages or login delays caused by host-name lookup failures
- Existing parameter, –U, revokes connections if host name is unavailable
  - The –g parameter is ignored if –U is specified
- Considerations: WHO will not show host name

```
>>--otelnetd--- ... ----+--------+---+--------+--- ... ---->< 
                        '-  -U  -'   '-  -g  -'
```


A new parameter,–g, is added to the otelnetd daemon to disable lookup of the client's host name from its IP address. You can enable this in your environment if host name lookup failures cause excessive error messages or excessive login delays.

The –g parameter is incompatible with the existing –U parameter, which revokes connections if the host name is unavailable. The new parameter is ignored if –U is specified.

You should note that if you specify the –g parameter, the WHO command will not show the host names of logged-in clients. This was true before in cases where the host name lookup failed, but if you specify –g, it is true for every client.

SNMP manager API trace level support

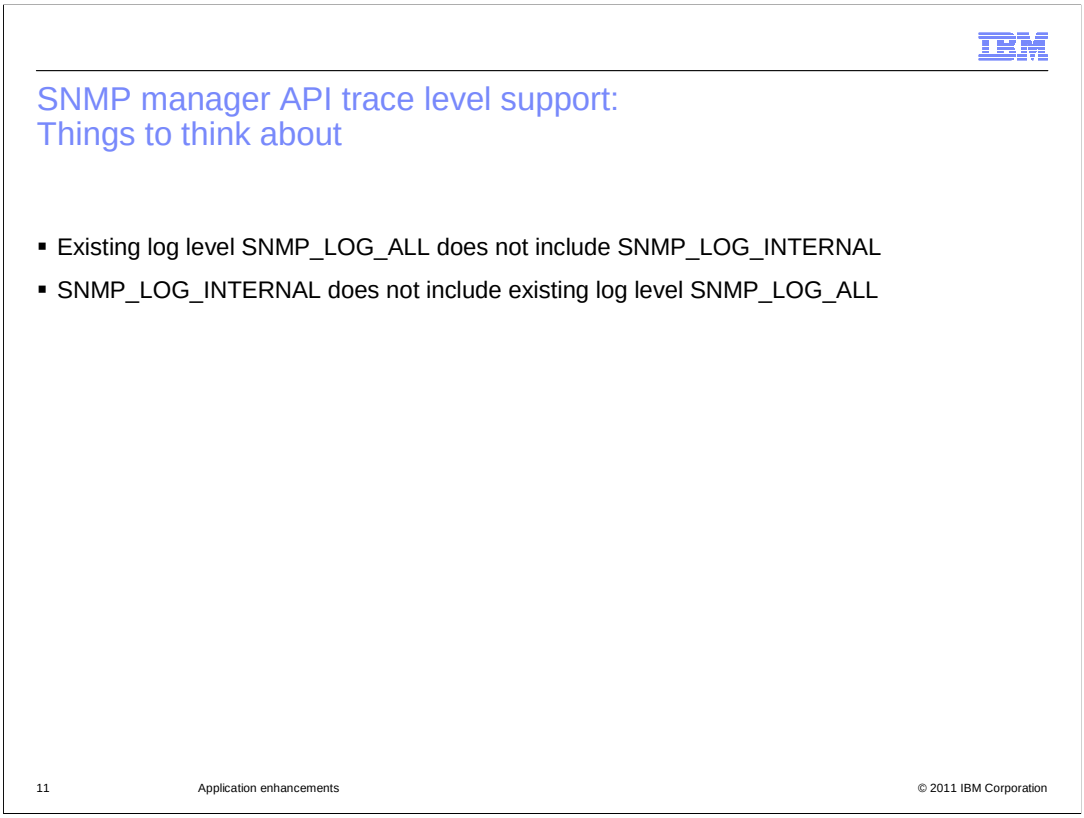
- SNMP manager API trace level is set by the API in two ways
  - *snmpSetLogLevel* function, using *logLevel* parameter
  - *snmpInitialize* function, using the value of environment variable SNMP_MGR_LOG_LEVEL
- New trace level SNMP_LOG_INTERNAL to specify tracing of packet processing
  - Facilitates diagnosis for certain problems
  - Packet processing traces are written to the output location specified by the syslogd configuration

There are two ways to set the trace level for the SNMP manager API. You can use the snmpSetLogLevel function in the SNMP Manager application, coding the required value for the logLevel parameter. Or you can set the environment variable SNMP_MGR_LOG_LEVEL to the required value before the SNMP manager application calls the snmpInitialize function. The snmpInitialize function uses the value of the environment variable to set the trace level. The trace level set by snmpInitialize using the environment variable overrides the value set by the snmpSetLogLevel function.

In z/OS V1R13, the logLevel has a new value, SNMP_LOG_INTERNAL.

The snmpInitialize function will also allow this new value to be specified using the SNMP_MGR_LOG_LEVEL environment variable.

Specification of this value, alone or in combination with any of the existing values, will trigger the logging of the packet processing traces under the SNMP Manager API.



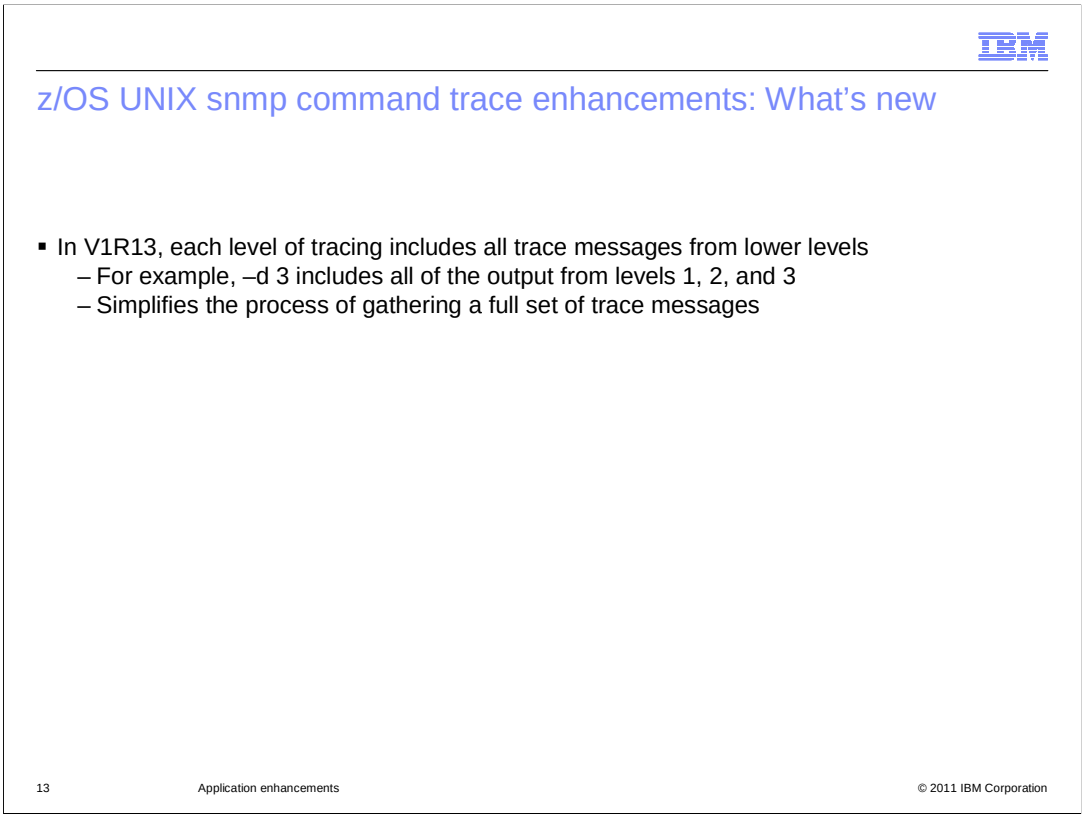
Existing log level SNMP_LOG_ALL continues to include all of the previously-existing log levels (SNMP_LOG_ERROR, SNMP_LOG_TRACE, SNMP_LOG_DUMP) but does not include SNMP_LOG_INTERNAL. This was done so that existing users of SNMP_LOG_ALL will not see unwanted SNMP_LOG_INTERNAL trace messages included in their output. Also, SNMP_LOG_INTERNAL does not include any of the previously-existing log levels (SNMP_LOG_ALL, SNMP_LOG_ERROR, SNMP_LOG_TRACE, or SNMP_LOG_DUMP).

z/OS UNIX snmp command trace enhancements background

- z/OS UNIX **snmp** command provides these SNMP manager functions
  - Query SNMP agents for network management information
  - Receive and format SNMP traps and notifications
- For debugging purposes, the command provides four levels of tracing, activated by specifying the *–d* parameter when invoking snmp
  - snmp **–d 4** <*other options*> <*command*> <*MIB object*>

The snmp command is provided as a very basic SNMP management application in the z/OS UNIX environment. The command is single threaded, and is intended primarily for testing the configuration of the SNMP environment. More robust managers are available, and examples of some IBM products include NetView® and Omegamon.

The four levels of tracing do have some overlap in what they output. However, not all of the trace messages output at one level are output at higher levels. Though this does not effect the functionality of the command itself, it can make it difficult to debug a problem. Instead of running the snmp command once to collect a single set of debug output, at times the command must be run multiple times with different levels of tracing in order to capture all of the needed information.



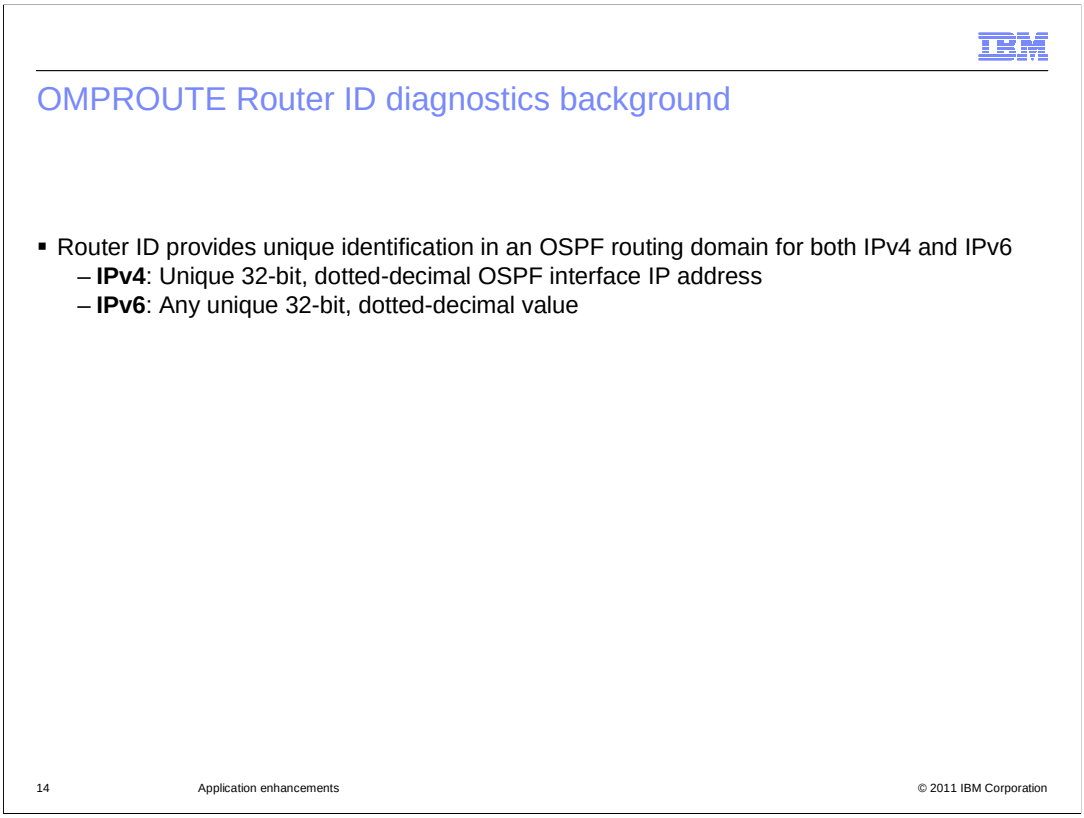
Now each level of tracing will incorporate all of the debug trace messages from the lower levels of tracing:

Level 2 contains all of the output from level 1 and level 2.

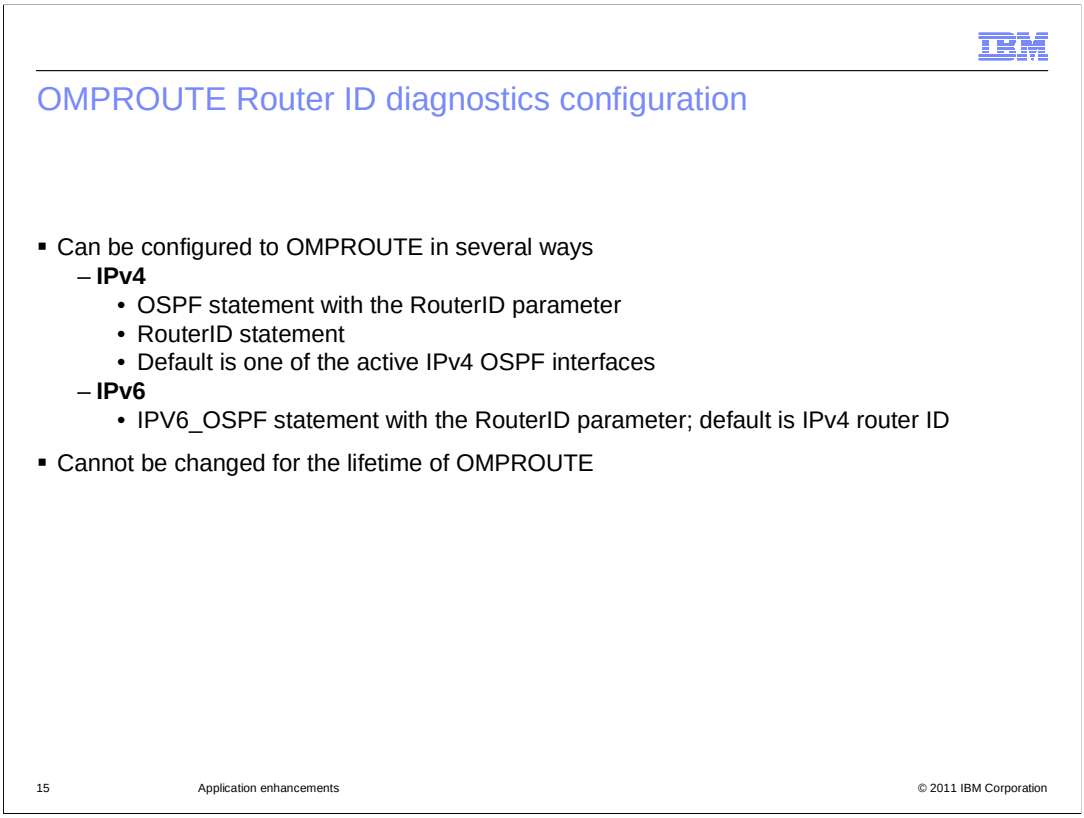
Level 3 contains all of the output from levels 1, 2, and 3.

Level 4 contains all of the output from levels 1, 2, 3, and 4.

The command syntax has not changed, and no action is required to take advantage of the enhancements. Although no new trace messages were added, tracing at a higher level might increase the total number of messages output than in prior releases. However, as the amount of output for snmp command tracing is already significantly large, the increase is relatively small.



A router ID is used by the OMPROUTE application as a 32-bit unique identifier within an OSPF autonomous system for both IPv4 and IPv6.



You configure an IPv4 router ID in the OMPROUTE configuration file on the RouterID statement or on the OSPF statement with the RouterID parameter.

You configure an IPv6 router ID in the OMPROUTE configuration file on the IPV6_OSPF statement with the RouterID parameter.

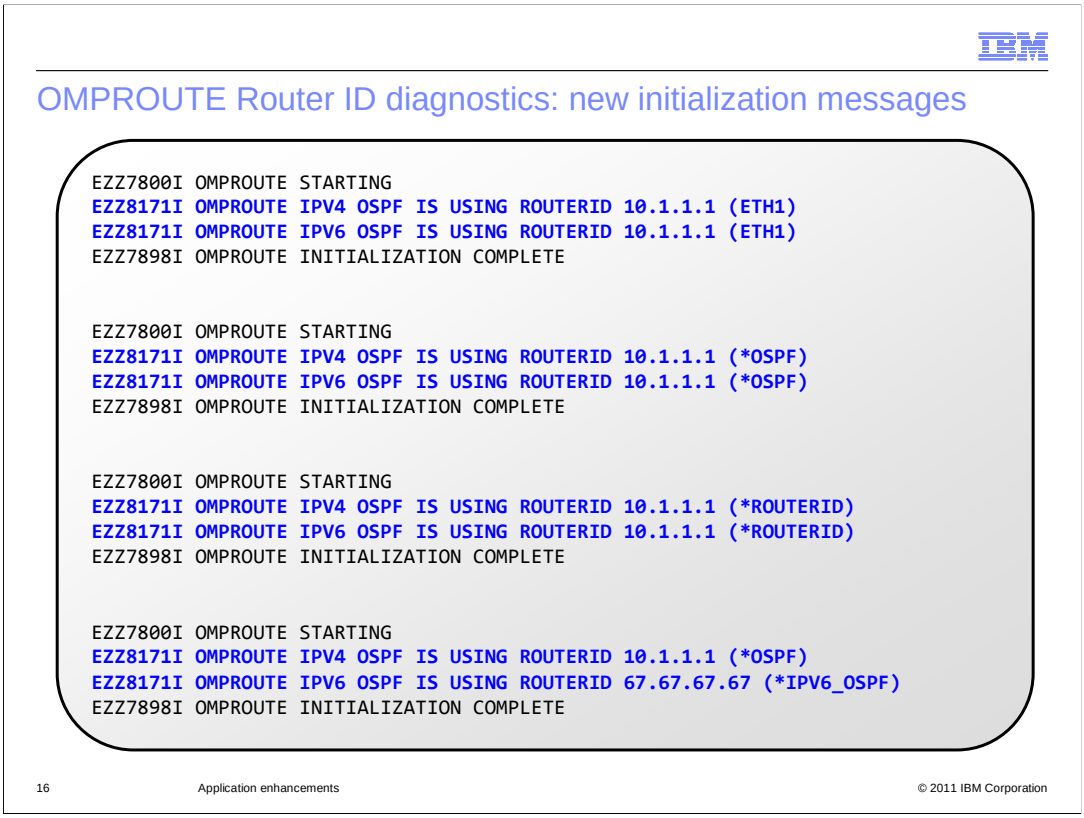
If multiple statements are coded for the IPv4 or IPv6 router ID, the last value is used.

If an IPv4 router ID is not specified in the OSPF configuration file, OMPROUTE chooses a home IPv4 address from the TCP/IP stack that matches one of the IPv4 OSPF interfaces for the router ID.

If using IPv6 OSPF and an IPv6 router ID is not configured, it defaults to the router ID used for IPv4 OSPF.

If IPv4 OSPF is not active, then an IPv6 router ID must be configured to activate IPv6 OSPF.

Once a router ID is in use, it cannot be changed while OMPROUTE is running. There is no dynamic reconfiguration support for the IPv4 router ID and limited dynamic reconfiguration support for the IPv6 router ID. A MODIFY OMPROUTE RECONFIG command can be used to add an IPv6 router ID if it was not previously defined or defaulted.



Here are examples of the new EZZ8171I message that displays the router IDs and their configuration sources during OMPROUTE initialization when using a configuration with both IPv4 and IPv6 OSPF enabled. When the router ID source is from a configuration statement, it is prefixed with an asterisk (*OSPF, *ROUTERID, *IPV6_OSPF). The absence of an asterisk indicates the router ID source is from an IPv4 OSPF interface.

The first set of EZZ8171I messages is issued when a default router ID has been selected by OMPROUTE. The configuration sources indicate that the router ID is from an IPv4 OSPF interface (ETH1).

The second set of EZZ8171I messages is issued when the RouterID parameter is specified on an IPv4 OSPF statement. OMPROUTE assigns this IPv4 router ID to the IPv6 router ID because there was no IPV6_OSPF statement with the RouterID parameter specified. The configuration sources indicate that the router ID is from the IPv4 OSPF statement (*OSPF).

The third set of EZZ8171I messages is issued when the router ID is specified on a RouterID statement. OMPROUTE assigns this IPv4 router ID to the IPv6 router ID because there was no IPV6_OSPF statement with the RouterID parameter specified. The configuration sources indicate that the router ID is from the RouterID statement (*ROUTERID).

The fourth set of EZZ8171I messages is issued when the RouterID parameters are specified on the IPv4 OSPF and IPV6_OSPF statements. The configuration sources indicate that the IPv4 router ID is from the IPv4 OSPF statement (*OSPF) and the IPv6 router ID is from the IPv6_OSPF statement (*IPV6_OSPF).

If only IPv4 OSPF or only IPv6 OSPF is enabled, then a single EZZ8171I message is issued indicating the router ID and its source.

OMPROUTE Router ID diagnostics: router ID source in displays

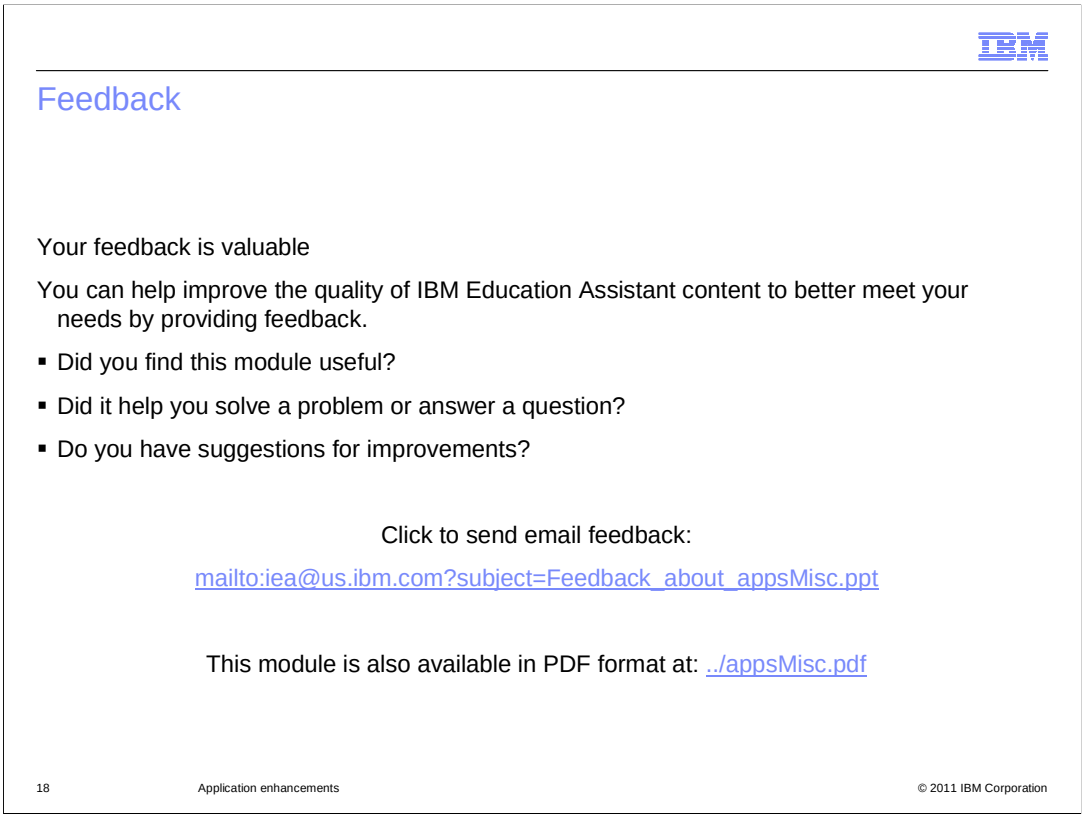
```
D TCPIP,,OMPROUTE,OSPF,STATISTICS
EZZ7856I OSPF STATISTICS
                OSPF ROUTER ID:         10.10.10.1 (ETH1)
                EXTERNAL COMPARISON:    TYPE 2
                AS BOUNDARY CAPABILITY: YES
                IMPORT EXTERNAL ROUTES: STA DIR SUB
                ORIG. DEFAULT ROUTE:    NO

. . .

D TCPIP,,OMPROUTE,IPV6OSPF,ALL
EZZ7970I IPV6 OSPF INFORMATION
TRACE6: 0, DEBUG6: 0
STACK AFFINITY            TCPCS
IPV6 OSPF PROTOCOL:       ENABLED
IPV6 OSPF ROUTER ID:      67.67.67.67 (*IPV6_OSPF)
DFLT IPV6 OSPF INST ID:   0

. . .
```


The IPv4 router ID source is now included in the DISPLAY TCPIP,OMPROUTE,OSPF,STATISTICS and MODIFY OMPROUTE,OSPF,STATISTICS reports. Similarly, the IPv6 router ID source is now included in the DISPLAY TCPIP,OMPROUTE,IPV6OSPF,ALL and MODIFY OMPROUTE,IPV6OSPF,ALL reports.



You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, disclaimer, and copyright information