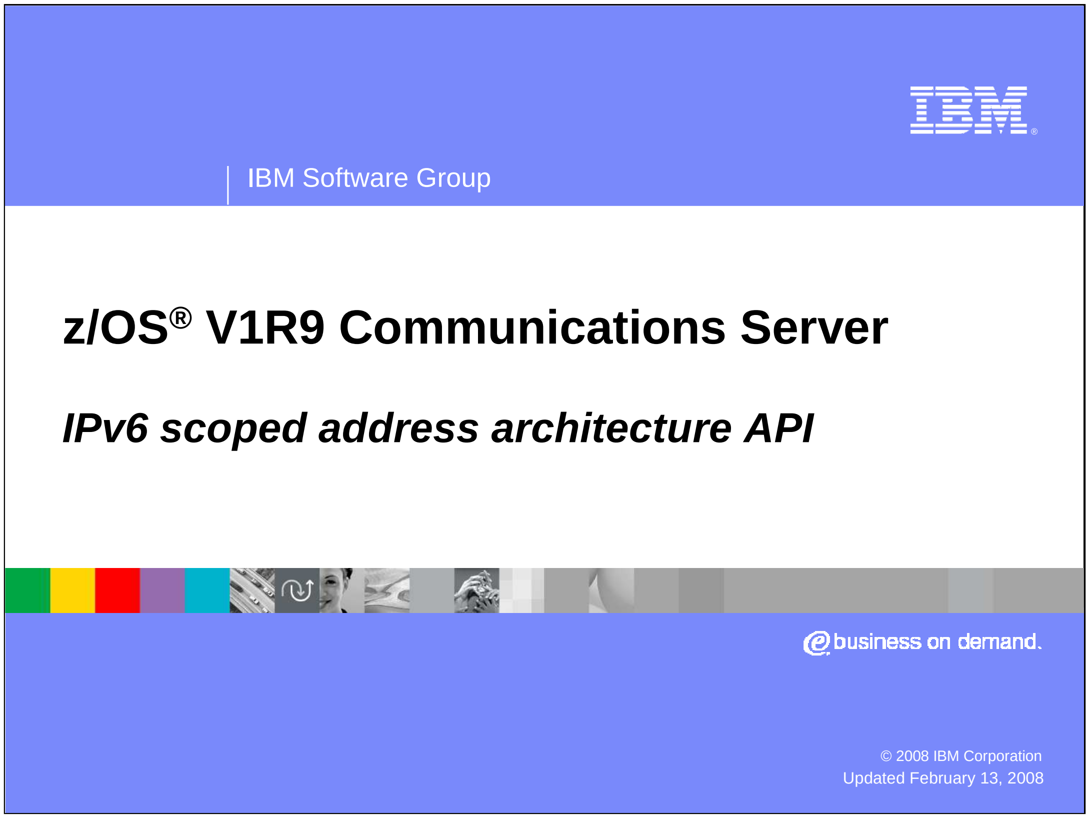


IBM Software Group

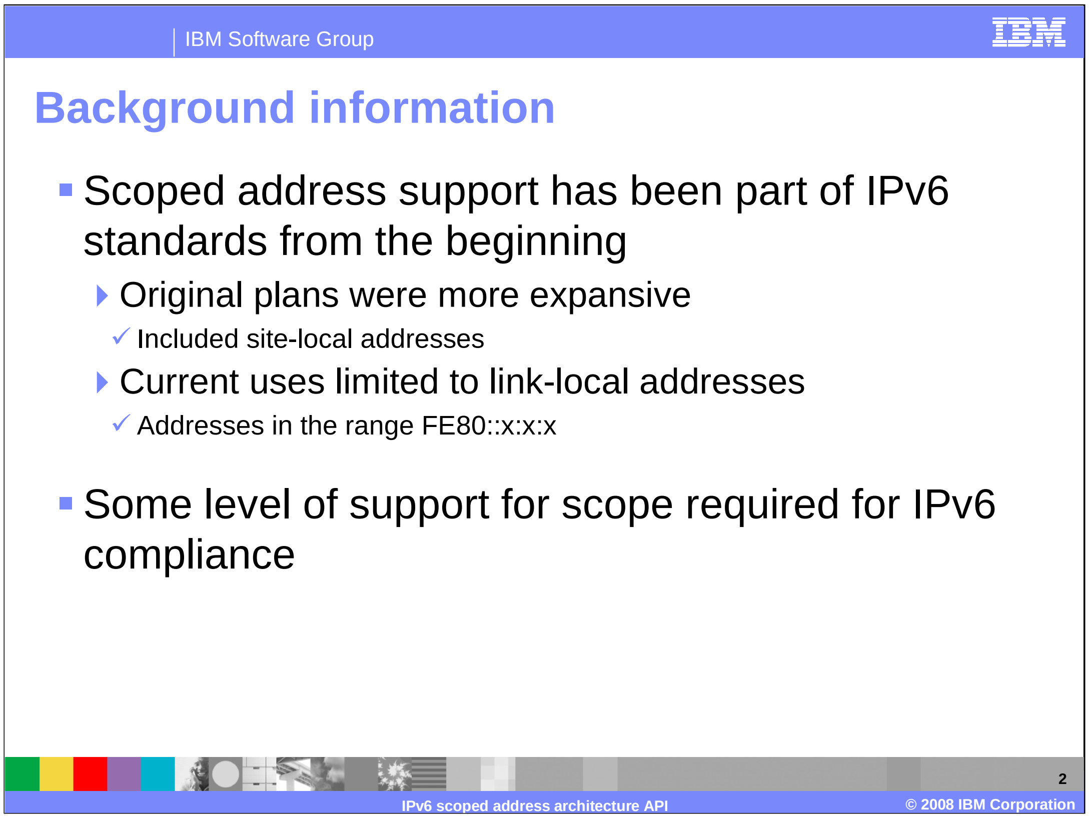
z/OS® V1R9 Communications Server

IPv6 scoped address architecture API

business on demand.

Updated February 13, 2008

This presentation discusses the support added to z/OS V1R9 Communications Server for the IPv6 scoped address API changes, primarily to the Resolver Getaddrinfo and Getnameinfo functions.



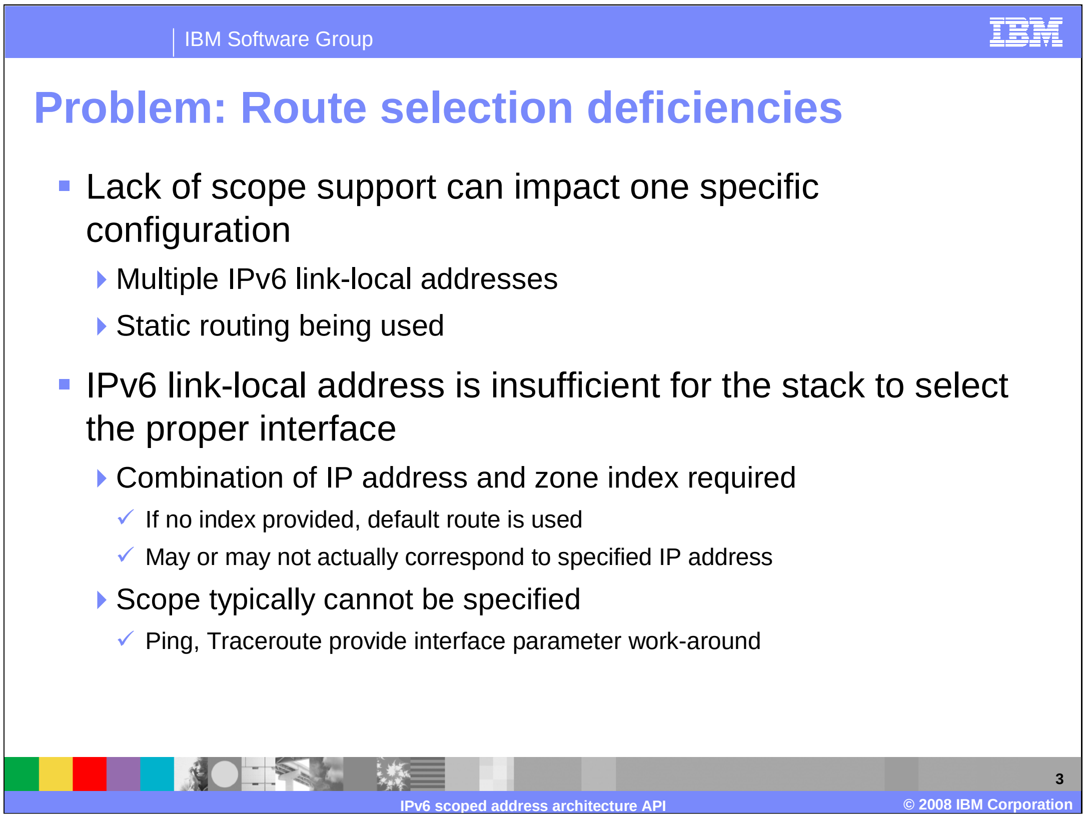
Background information

- Scoped address support has been part of IPv6 standards from the beginning
  - Original plans were more expansive
    - ✓ Included site-local addresses
  - Current uses limited to link-local addresses
    - ✓ Addresses in the range FE80::x:x:x
- Some level of support for scope required for IPv6 compliance

IPv6 has always had the concept of scoped addresses, but z/OS has never fully supported the function. Part of that was due to the cost of fully implementing support for scoped addresses (sometimes also referred to as zones), but also because the concept, while present, had not been fully defined. For instance, at one point scoped addresses included a set of addresses defined as site-local addresses, but that class of addresses has since been downplayed by the IPv6 standards. The only addresses left for which the concept of scope now applies are link-local addresses.

Link-local addresses are addresses that refer to only a particular physical link, or the physical network directly attached to that link (for example, LAN). They are used only for local communication on that physical link, and routers are designed to not forward datagrams that use link-local addresses. Link-local addresses are typically dynamically assigned by the TCP/IP stack, and are mostly used for so-called “bootstrap” functions or diagnostic purposes.

In order to maintain compliance with IPv6 standards, z/OS needs to implement some additional level of scoped address support. The level chosen could range from full implementation of zones, to recognizing and utilizing scope information on the various z/OS applications and APIs.



IBM Software Group

Problem: Route selection deficiencies

- Lack of scope support can impact one specific configuration
  - Multiple IPv6 link-local addresses
  - Static routing being used
- IPv6 link-local address is insufficient for the stack to select the proper interface
  - Combination of IP address and zone index required
    - ✓ If no index provided, default route is used
    - ✓ May or may not actually correspond to specified IP address
  - Scope typically cannot be specified
    - ✓ Ping, Traceroute provide interface parameter work-around

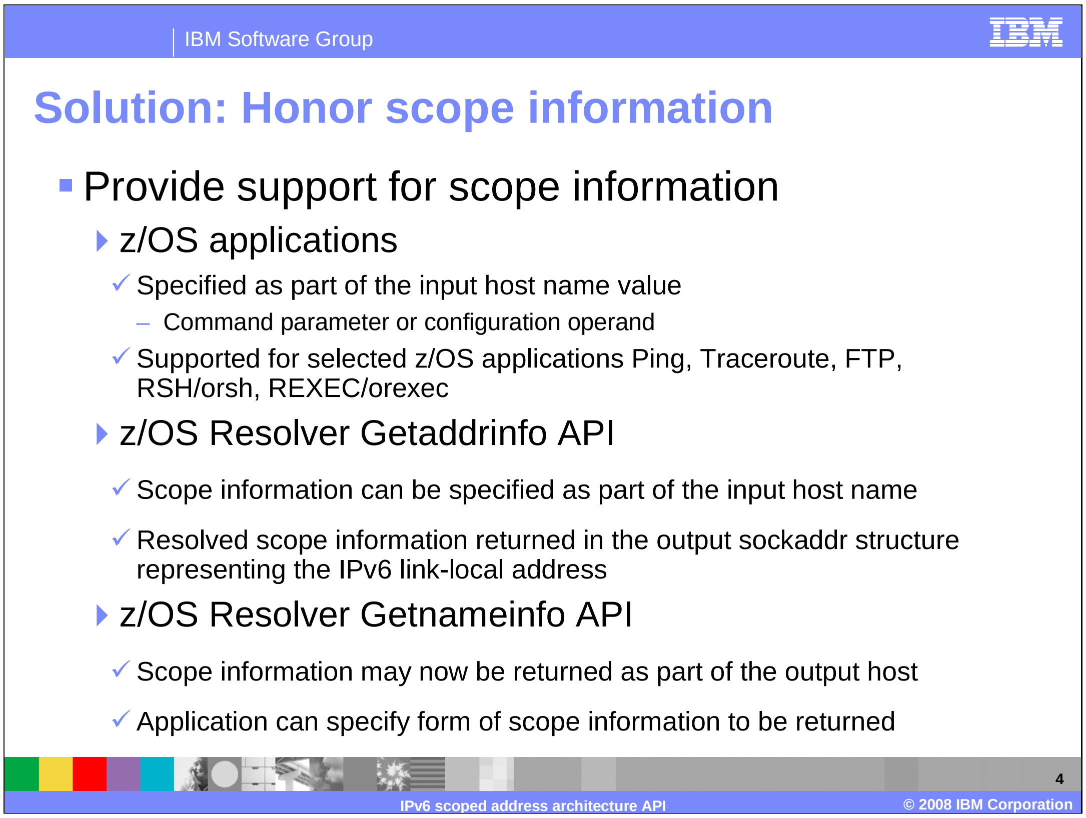
3

IPv6 scoped address architecture API

There is one configuration in particular where the absence of support for scoped addresses, at the API level, could impact a z/OS user. The situation would involve a configuration where multiple IPv6 link-local addresses have been dynamically assigned. To further complicate matters, static routing is being used in the configuration. There are other possibilities where even the use of dynamic routing could lead to complications, but the more likely situation is that static routing is being used.

In such a situation, to successful route data over a given IPv6 link-local address, both the address and the zone index value need to be specified. The zone index is a value assigned by the stack to represent the correct entry (or interface) in the routing table. If the zone index is not present, then the stack uses the “default route” for this configuration. If the default route uses the interface that matches the IPv6 link-local address that was specified, everything works just fine. If, however, the default route does not use the correct interface for the specified IPv6 link-local address, then a routing error is encountered and the application request fails or times out.

Some applications, such as Ping and Traceroute, have had parameters defined that allow them to work around this problem, but other applications such as FTP have no such work-around. Ideally, here would be a standard mechanism in place to allow the z/OS user or the application to specify the proper interface to use with link-local addresses.



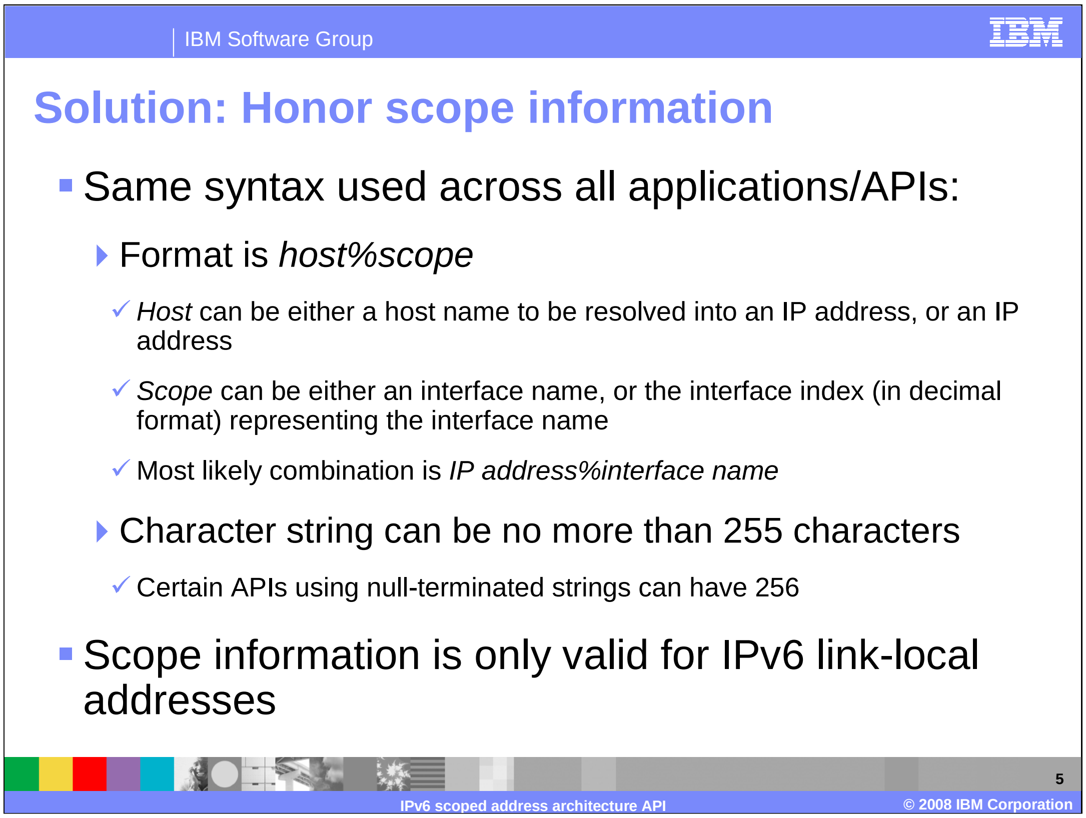
Solution: Honor scope information

- Provide support for scope information
  - z/OS applications
    - ✓ Specified as part of the input host name value
      - Command parameter or configuration operand
    - ✓ Supported for selected z/OS applications Ping, Traceroute, FTP, RSH/orsh, REXEC/orexec
  - z/OS Resolver Getaddrinfo API
    - ✓ Scope information can be specified as part of the input host name
    - ✓ Resolved scope information returned in the output sockaddr structure representing the IPv6 link-local address
  - z/OS Resolver Getnameinfo API
    - ✓ Scope information may now be returned as part of the output host
    - ✓ Application can specify form of scope information to be returned

The generalized solution is to permit scope information to be present as part of the host name parameter or configuration operand. This support is extended to a subset of the z/OS applications listed on the slide. The z/OS Resolver was also updated to process the scope information correctly.

The underlying basis for providing support of scope information is the z/OS Resolver updates for Getaddrinfo, since the z/OS applications that support scope information will issue Getaddrinfo in the background to manipulate the scope information. The scope information will only be processed by Getaddrinfo when the input host name is either an IPv6 link-local address, or when the host name provided resolves into one or more IPv6 link-local addresses. Be careful with the latter situation however, especially if for some reason you have configured one host name to represent multiple link-local addresses. The Resolver has no way of determining which of the resolved link-local addresses really is represented by the input scope information, so by default the Resolver will apply the scope information to EVERY resolved link-local address. Typically this will result in only one output sockaddr structure being correct, so it is best to stick to one link-local address per host name, if you even bother to use host names of link-local addresses. Assuming there is an IPv6 link-local address in play, the Resolver will handle the scope information in one of two ways: If the input scope information is an interface name, then the Resolver issues a system IOCTL to acquire the routing table. A simple lookup is performed to find the specified interface name, and the corresponding interface index value is returned as the *sin6_scope_id* value in the output sockaddr structure. If the input scope information is an interface index, the Resolver will perform a sanity check to ensure that the index works on this system. The same IOCTL is issued to get the same routing table, but the lookup is performed to find the index value in the table, not the name. If the index is present, the input value is echoed in the sockaddr structure as described above. This sanity check is performed to ensure, as much as possible, that any sockaddr structure returned by the Resolver to the user is valid for use in establishing a connection or for sending data to the target host. If the Resolver attempts resolution but the lookup fails, the Resolver call fails.

The other Resolver API that manipulates scope information is Getnameinfo. Getnameinfo processing will take the scope information in the input sockaddr structure, namely the *sin6_scope_id* field, and append it to the end of the output host name value. The



IBM Software Group

Solution: Honor scope information

- Same syntax used across all applications/APIs:
  - Format is *host%scope*
    - ✓ *Host* can be either a host name to be resolved into an IP address, or an IP address
    - ✓ *Scope* can be either an interface name, or the interface index (in decimal format) representing the interface name
    - ✓ Most likely combination is *IP address%interface name*
  - Character string can be no more than 255 characters
    - ✓ Certain APIs using null-terminated strings can have 256
- Scope information is only valid for IPv6 link-local addresses

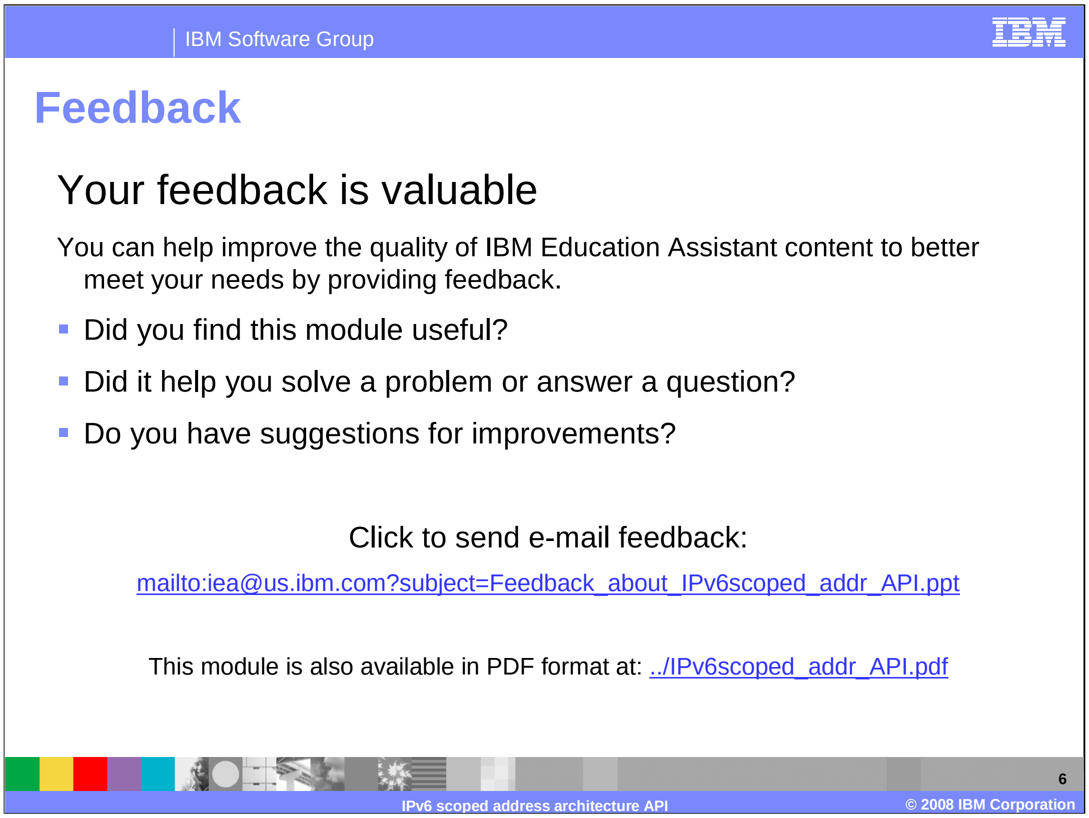
5

IPv6 scoped address architecture API

In any instance where scope information is permitted to be specified, the same syntax is used. The scope information is appended to the host name, with a percent sign used as the delimiter value. Note that “host name” in this discussion, means one of two things: An actual host name defined to DNS that maps to an IP address or an IP address. The IP address specified, or resolved to using the host name, must be an IPv6 link-local address, or scope information is meaningless. Likewise, in this discussion, “scope information” can take on two distinct forms: The name assigned to the interface (physical link) by the user on the INTERFACE statement or the zone index assigned to the interface by the stack. The value must be specified in decimal form, not hexadecimal. While any combination of host name or IP address with interface name or interface index is permitted, in general, the most likely choice would be IP address with interface name. In most situations, link-local addresses would not have a host name assigned to them, leaving the IP address as the only choice. The zone index value for a given interface can only be acquired programmatically, not through operator displays, and in any event can change for an interface from one TCP/IP stack activation to the next. The interface name is likely to be constant and can be displayed (along with the associated link-local address) using Netstat commands, so interface name is much more accessible than the zone index.

In order to minimize the impact of the addition of scope information to the z/OS APIs, the existing restriction of 255 characters (or 256, for APIs that use null-termination characters) for a “host name” has been maintained. This was not believed to be a concern because (a) most host names are far less than 255 characters long and (b) IPv6 link-local addresses typically would not have a host name assigned to them anyway, since they are dynamically assigned to an interface by the stack.

The full range of z/OS IPv6 capable APIs that provide support for Getaddrinfo and Getnameinfo calls are capable of handling scope information.



Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

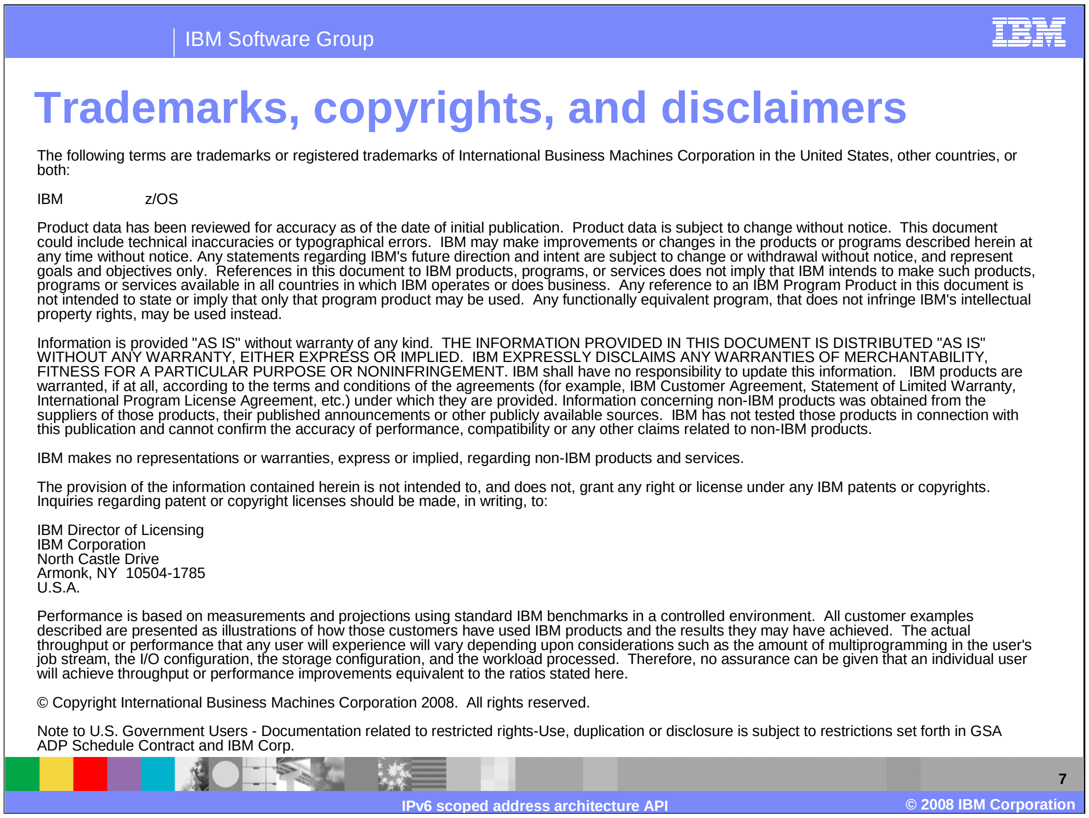
- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_IPv6scoped_addr_API.ppt

This module is also available in PDF format at: ../IPv6scoped_addr_API.pdf

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



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

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.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.