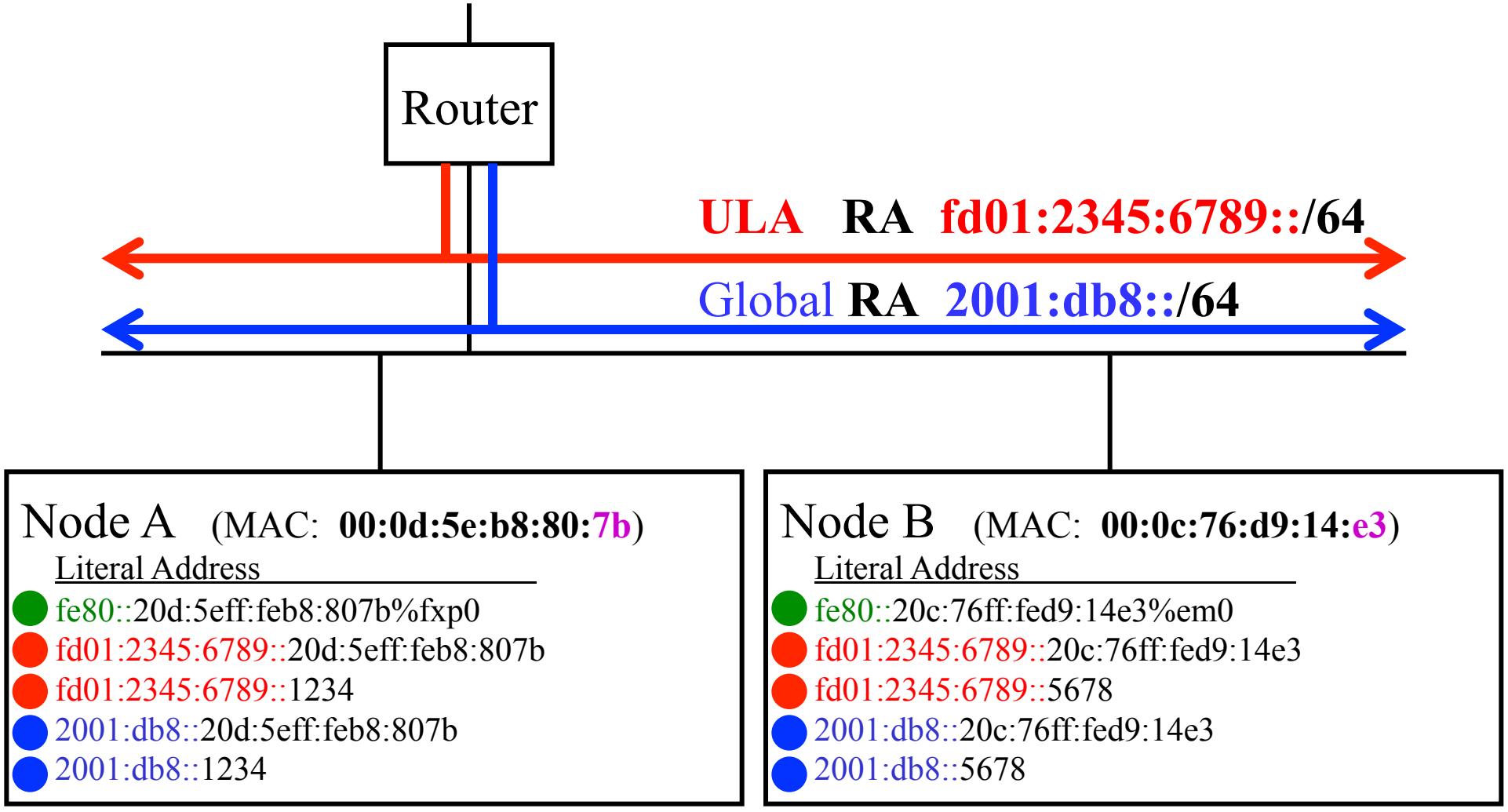
Corresponding Auto Names for IPv6 Addresses

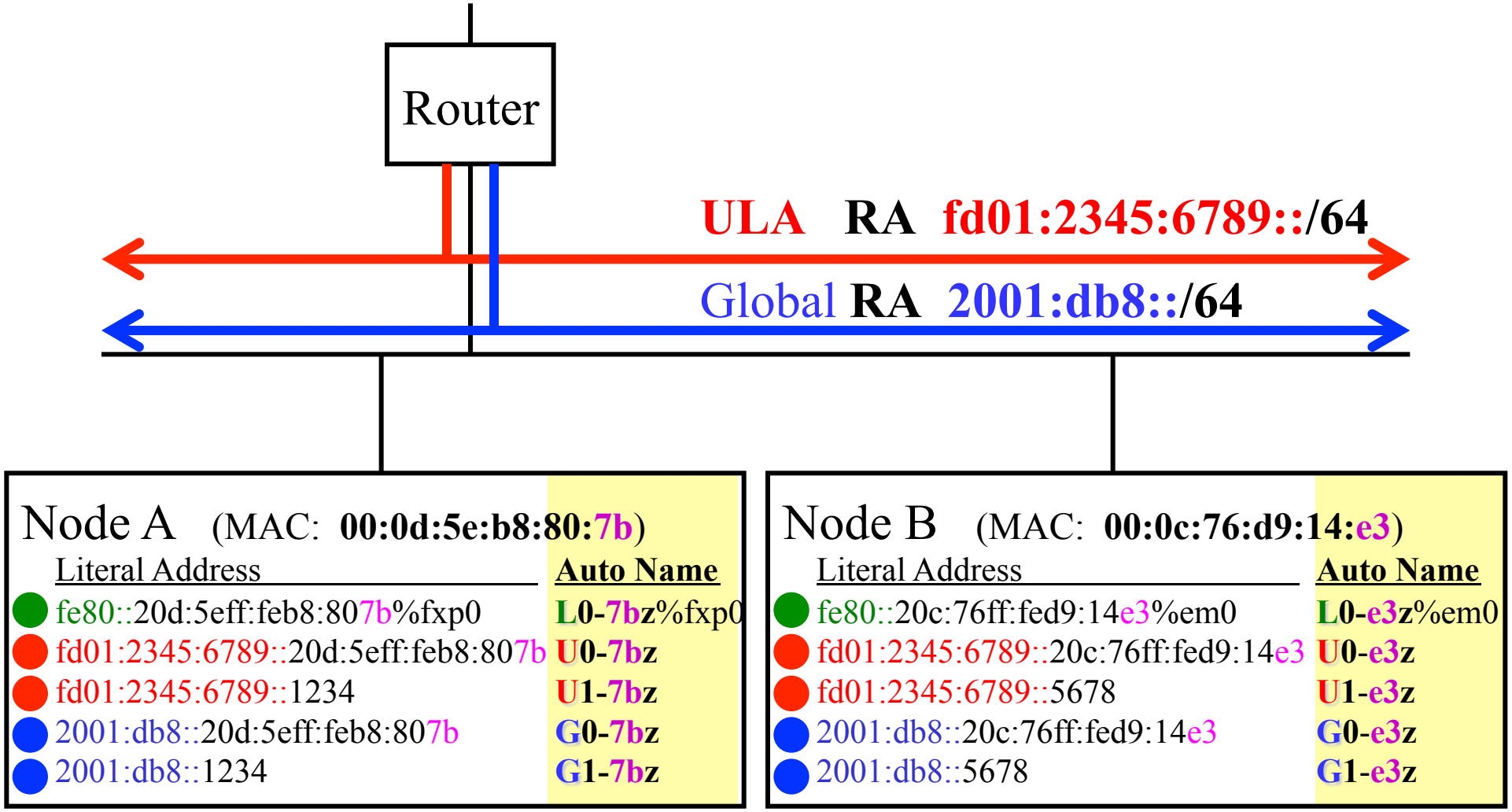
<draft-kitamura-ipv6-auto-name-03.txt>

Hiroshi KITAMURA
NEC Corporation
kitamura@da.jp.nec.com

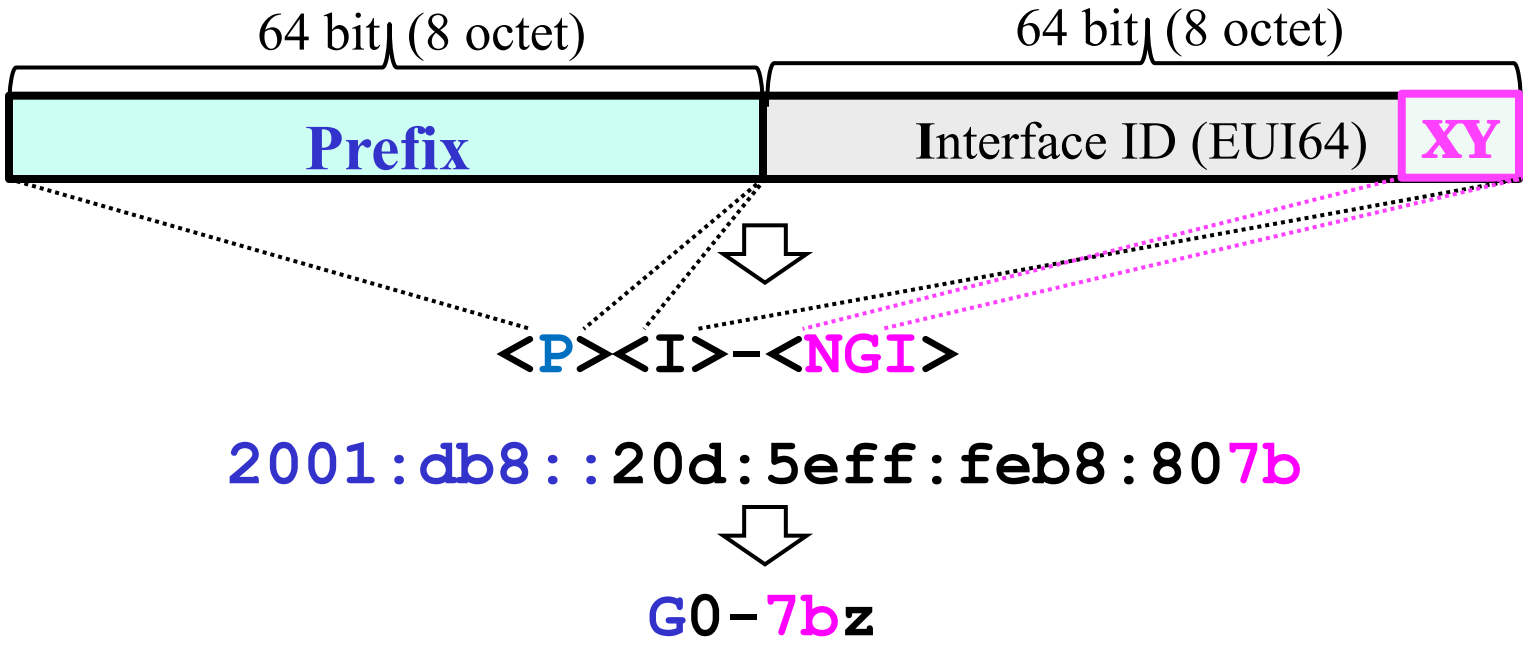
Assumed Typical IPv6 Communication Environment



Auto Names Examples



Literal Address ⇨ Auto Name



<P>: **P**refix part
1 character: (e.g., 'L', 'U', 'G')

<I>: **I**nterface ID part
1 character: (e.g., '0', '1', '2', , , '9', 'a', , , 'z')

<NGI>: **N**ode (Interface) **G**roup **I**D
3 characters: (e.g., '7bz', '3ez') **inherited** from the **last octet** (2 characters) of the node's MAC address + 1 char. for collision avoidance (usually 'z')

Auto Names techniques in short

- Under certain **scoped name** environment,
  All IPv6 addresses (formed as Prefix + I/F ID) are
  shown in only ***fixed 6 characters***
  ("<P><I>-<NGI>“) strings format.
  [kind of address **compression techniques** are used.]
- IPv6 Address information is annotated and changed
  almost meaningless ⇨ **meaningful**

- Human can remember, understand and ‘type’
  Auto Names (instead of literal IPv6 addresses).

Example 1: (Wireshark packet dump)
Easy Compare / Distinguish IPv6 addresses

It is very difficult to compare (similar) literal IPv6 addresses

Source	Destination	Protocol	Info
2001:dc2a:cafe:babe:202:b3ff:fe4d:1522	2001:dc2a:cafe:babe:202:b3ff:fe4d:15a2	TCP	52456 > ssh [SYN] Seq=(
2001:dc2a:cafe:babe:202:b3ff:fe4d:15a2	2001:dc2a:cafe:babe:202:b3ff:fe4d:1522	TCP	ssh > 52456 [SYN, ACK]
2001:dc2a:cafe:babe:202:b3ff:fe4d:1522	2001:dc2a:cafe:babe:202:b3ff:fe4d:15a2	TCP	52456 > ssh [ACK] Seq=1
2001:dc2a:cafe:babe:202:b3ff:fe4d:15a2	2001:dc2a:cafe:babe:202:b3ff:fe4d:1522	SSHv2	Server Protocol: SSH-2.
2001:dc2a:cafe:babe:202:b3ff:fe4d:1522	2001:dc2a:cafe:babe:202:b3ff:fe4d:15a2	SSHv2	Client Protocol: SSH-2.
10.2.0.55	10.2.0.225	ICMP	Echo (ping) reply ic

Source	Destination	Protocol	Info
G0-22z	G0-a2z	TCP	52456 > ssh [SYN] Seq=(
G0-a2z	G0-22z	TCP	ssh > 52456 [SYN, ACK]
G0-22z	G0-a2z	TCP	52456 > ssh [ACK] Seq=1
G0-a2z	G0-22z	SSHv2	Server Protocol: SSH-2.
G0-22z	G0-a2z	SSHv2	Client Protocol: SSH-2.
10.2.0.55	10.2.0.225	ICMP	Echo (ping) reply ic

It becomes easy to compare Auto Names (fixed 6 char. Strings)

Example 2: (Wireshark packet dump)
Easy Identify / Group IPv6 addresses

It is very difficult to identify literal IPv6 addresses (set to the same node)

Source	Destination	Protocol	Info
10.2.0.225	10.2.0.55	ICMP	Echo (ping) request id
2001:dd2a:cafe:babe:202:b3ff:fe4d:1522	2001:dd2a:cafe:babe:202:b3ff:fe4d:15ab	ICMPv6	Echo (ping) request id=
fd02:dc2a:cafe:babe:1f31:beef:face:fa1e	fd02:dc2a:cafe:babe:1f32:beef:face:fa1e	ICMPv6	Echo (ping) request id=
fe80::202:b3ff:fe4d:1522	fe80::202:b3ff:fe4d:15ab	ICMPv6	Echo (ping) request id=
2001:dd2a:cafe:babe::68	2001:dd2a:cafe:babe::69	ICMPv6	Neighbor Solicitation f
fd02:dc2a:cafe:babe:202:b3ff:fe4d:1522	fd02:dc2a:cafe:babe:202:b3ff:fe4d:15ab	ICMPv6	Neighbor Advertisement
10.2.0.55	10.2.0.225	ICMP	Echo (ping) reply id
2001:dd2a:cafe:babe:202:b3ff:fe4d:15ab	2001:dd2a:cafe:babe:202:b3ff:fe4d:1522	ICMPv6	Echo (ping) reply id=0x
fd02:dc2a:cafe:babe:1f32:beef:face:fa1e	fd02:dc2a:cafe:babe:1f31:beef:face:fa1e	ICMPv6	Neighbor Advertisement
fe80::202:b3ff:fe4d:15ab	fe80::202:b3ff:fe4d:1522	ICMPv6	Echo (ping) reply id=0x
2001:dd2a:cafe:babe::69	2001:dd2a:cafe:babe::68	ICMPv6	Echo (ping) reply id=0x
fd02:dc2a:cafe:babe:202:b3ff:fe4d:15ab	fd02:dc2a:cafe:babe:202:b3ff:fe4d:1522	ICMPv6	Echo (ping) reply id=0x

⇩

Source	Destination	Protocol	Info
10.2.0.225	10.2.0.55	ICMP	Echo (ping) request id
G0-22z	G0_abz	ICMPv6	Echo (ping) request id=
Ua_22z	Ua_abz	ICMPv6	Echo (ping) request id=
L0-22z	L0_abz	ICMPv6	Echo (ping) request id=
G1_22z	G1_abz	ICMPv6	Neighbor Solicitation f
U0-22z	U0_abz	ICMPv6	Neighbor Advertisement
10.2.0.55	10.2.0.225	ICMP	Echo (ping) reply id
G0_abz	G0-22z	ICMPv6	Echo (ping) reply id=0x
Ua_abz	Ua_22z	ICMPv6	Neighbor Advertisement
L0_abz	L0-22z	ICMPv6	Echo (ping) reply id=0x
G1_abz	G1_22z	ICMPv6	Echo (ping) reply id=0x
U0_abz	U0-22z	ICMPv6	Echo (ping) reply id=0x

Deployed Notions and Functions used in **Auto Names**

- **Stateless** Name

	Stateful	**Stateless**
Address	DHCPv6	SLAAC
Name	Existing Domain Names	**Auto Names**

- **Scoped** Name

	Global	Site-Local (ULA)	**Link-Local**	**Node-Local**
Address	e.g., 2001:db8::/64	e.g., fd01:2345:6789::/64	fe80::/64	
Name	Existing Domain Names	Existing Domain Names / **Auto Names**	**Auto Names**	**Auto Names**

Scope is dependent on how **Auto Names** data is dealt and which "name services" are used.

Name Services

- Various types of '**name services**' can be used for Auto Names.
  - **DNS** can be used for **wide scoped Auto Names**.
    - All OS have DNS resolver implementations.
    - By using the DNS user-authenticate implementation, it is easy to achieve the 'Scoped Name' features.
  - "**/etc/hosts**" can be used for **narrow scoped Auto Names**. (especially for Link-local scoped ones)
    - It is very easy to deploy Auto Names.

Questions on Auto Names Design

<P>: No need to ask: because it is minimum 1 char.
<I>: No need to ask: because it is minimum 1 char.

<NGI>: **Ask to Audience**:

A: **3** char. (current design)
Inherited from the **last octet** (2char.) of MAC
+ 1 char. for collision avoidance (usually 'z')

B: **2** char.
Not collided Random 2 char. or
Inherited 1 char. + 1 char. for collision avoidance.

C: **1** char.
Not collided Random 1 char.

Auto Names <NGI> Design Analysis

Type:	Merit	Demerit
A: **3** char. (**current design**)	**Easy to remember**, because 2 chars are inherited from MAC address.	3 char. is rather long. Need to remember 1 char. for collision avoidance, (but it is usually 'z')
B: **2** char. (got suggestion)	2 char. is shorter than 3 char.	Need to remember at least 1 random char. for collision avoidance
C: **1** char.	1 char. is **short enough**	Capacity is small.

Please let us know your preference.

Discussions

Please let us know your comments.

Goal of this I-D is
to be published as “Informational RFC”.

Reserved slides are started from here.

Site-dependent Mapping tables (for ***collision avoidance***)

Mapping tables are used **only when Auto Names** are **generated**

(These tables are **not used** for usual name resolving operations)

- MAC address – <NGI> value mapping table

MAC Address	<NGI> value
00:0d:5e:b8:80:7b	**-7bz**
00:0c:76:d9:14:e3	**-e3z**

- Prefix – <P> value mapping table

Prefix		<P> value
fe80::/64	Link-Local	**L**
fd01:2345:6789::/64	Site-Local (ULA)	**U**
2001:db8::/64	Global	**G**

<I> Value

<I> value stands for Interface ID part of IPv6 Address
as 1 character format.

<I> value assignment is based on
three address type categorization

type	description
"0"	used for **EUI64-based** address
"1" - "9"	used for **manually** set addresses (stateful addresses will be categorized here)
"a" - "z"	used for **automatically** generated and set addresses except EUI64-based (Temporary addresses are categorized here)

Address Type Distinction

- EUI64-based Address Identification
  - When **IPv6 and MAC** addresses are found **simultaneously**, it is easy to identify.

- Manual or Automatic Distinction
  - **Human bias** is checked by using "**Zero Contain Rate**" in IPv6 Address.

<NGI> Value

<NGI> value is also called Auto **Name-Suffix**.

<NGI> value is shown as 'XYZ' format:

'XY': (1st, 2nd chars) are **inherited** from
the **last octet** (2 characters) of the node's MAC address

'Z' : (3rd char) suffix char to **avoid a collision** of 'XY'
starting from "z"
if 'XY' is collided, 'Z' is changed into "y", "x" „,

Collision Probability of 256 states (1 octet):

By using the ***birthday paradox theorem***, probability is calculated.
If there are **19 nodes** (interfaces) on the same scope,
collision is happened **with 50%** probability.

Collision check procedure for 'XY' is necessary.