Chapter 18

NIC Resolution Process

This chapter provides information about the NIC resolution process. You should be familiar with this information if you customize a NIC configuration scenario. Topics include:

Overview of the Resolution Process

Because NIC can process all types of network data, you must use different resolution processes for different types of data mappings to maximize the performance of the NIC configuration. Resolving data requests consumes significant resources.

Table 16 shows the resolutions that the components in the NIC configuration scenarios perform. For customized types of resolutions, contact Juniper Networks Professional Services.

Table 16: Available NIC Resolutions

Key	Value
Subscriber's IP address (JUNOS routing platform)	SAE reference
Subscriber's IP address	Subscriber's login name
Subscriber's IP address	SAE reference
Subscriber's login name	SAE reference
Subscriber's username	SAE reference
Access DN	SAE reference

NIC Realms

Each resolution process and the resolvers that perform that process are defined by a *realm*—a group of resolvers that perform a series of resolution tasks to provide a mapping from a specified key to a specified data type. For example, the sample data provided for the NIC includes a realm called dn in which the resolution process takes an access subscriber's distinguished name (DN) as the key and returns a reference to the SAE managing this subscriber as the value.

A set of hosts in a NIC can support multiple realms. Similarly, the agents in a NIC can support more than one realm. However, you can assign a resolver only to one realm.

A NIC host can support NIC resolvers for multiple realms. Consequently, you can simplify the NIC configuration and minimize the use of network resources by limiting the number of NIC hosts in your NIC configuration. NIC hosts can also handle multiple NIC resolvers in the same realm. In this case, when a NIC host receives a request, it chooses a NIC resolver as follows:

1. It identifies the NIC resolvers that are available to process the request.

2. If multiple NIC resolvers are available, it obtains a cost value associated with the resolution process from each resolver and selects the resolver that has the lowest cost value.

Key to Value Resolution

A resolution process typically defines several transitions or *roles*, with each transition resolving a NIC key to a value. For example, the resolution process to identify the SAE that manages a particular subscriber based on that subscriber's IP address involves the following roles:

1. Given the IP address, determine the IP address pool.

2. From the IP address pool, determine the VR.

3. From the VR, determine the SAE that manages that VR.

A role specifies the types of data with which it works. NIC supports a number of data types, including one that lets you add an identifier to other data types to let you specify different values for one data type.

For information about NIC data types, see *NIC Data Types* on page 285 and *Constraints as NIC Data Types* on page 287.

NIC Data Types

The NIC supports the data types that appear in the following list. You can qualify these data types by adding an identifier to:

- Distinguish between different instances of a data type in a resolution scenario.
- Provide information about a data type to clarify the use of that data type in a resolution.

For information about qualifying data types, see *Chapter 19, Customizing a NIC Configuration*.

AnyString

- Generic data type to represent the information that you want to collect.
- Value—Alphanumeric characters
- Guidelines—You can qualify this data type with an identifier to provide information about the type of data that AnyString represents.
- Example—My(IP), My(Vr)

Dn

- DN of an access.
- Value—DN
- Example—*accessName = PrimaryAccess, enterpriseName = juniper, ou = Sunnyvale, retailerName = VPNprovider, o = Users, o = umc*

Domain

- Domain name.
- Value—Name of a domain
- Example—Example.net

Enterprise

- DN of an enterprise.
- Value—DN
- Example—*enterpriseName = juniper, ou = Sunnyvale, retailerName = VPNprovider, o = Users, o = umc*

Router

- Name of router.
- Value—Text string
- Example—router1

Interface

- Name of a router's interface.
- Value— < interfaceName > @ < vrName > @ < routerName >
  - < interfaceName > —Name of the interface, such as fastEthernet 3/1, exactly as it is configured on the router
  - < vrName > —Name of VR exactly as it is configured on the router
  - < routerName > —Name of router exactly as it is configured on the router
- Example—FastEthernet4/1.0@boston@router1

InterfaceId

- Identifier of an interface.
- Value— < intfIndex > @ < routerName >
  - < intfIndex > —Simple Network Management Protocol (SNMP) index of the interface
  - < routerName > —Name of router exactly as it is configured on the router
- Example—4@router1

Ip

- Subscriber's IP address.
- Value—IP address
- Example—192.0.2.10

IpPool

- IP address pool.
- Value—Range of IP addresses enclosed in square brackets and parentheses
- Guidelines—If you enter an IP address that includes a value greater than 255 in one octet of the address, that part of the address is masked to fit the eight bits.
- Example—([192.0.2.0 192.0.2.255])

SaeId

- SAE reference.
- Value—CORBA interoperable object reference (IOR) for SAE
- Example—IOR:000000000000002438444C3A736...

Vr

- Name of the virtual router.
- Value— < vrName > @ < routerName >
  - < vrName > —Name of VR exactly as it is configured on the router
  - < routerName > —Name of router exactly as it is configured on the router
- Example—vr1@router1

Constraints as NIC Data Types

Constraints are data types that a resolver uses when it executes a role. You can define:

- Multiple constraints for a role—Software performs an OR operation to determine whether the constraint is met.
- Multiple data types in a constraint—Software performs an AND operation to determine whether the multiple constraints are met.

Constraints can be either mandatory or optional. If a constraint is mandatory and the resolver for the role does not receive an appropriate value in the data request, the resolver must obtain the constraint value from other NIC resolvers. However, if a constraint is optional and the resolver for the role does not receive an appropriate value in the data request, the resolver can execute its role without the constraint value. In this case, the resolver may obtain multiple values for the data key, and the NIC host responds to the NIC proxy as follows:

- If the request is for multiple results, the host provides all the results.
- If the request is for one result and the resolution process returns different results, the host returns an error message.
- If the resolution process returns multiple instances of the same result, the resolver provides only one result.

For example, if you want to obtain an SAE reference for a subscriber's IP address, you could define the following roles:

1. From the IP address, determine the VR (mandatory constraint IpPool).
2. From the VR, determine the SAE that manages that VR.

Because the first step has a mandatory constraint, the resolver for this role must use the IP pool supplied in the request, or obtain the IP pool from another resolver that determines IP pools from IP addresses. So you must define an extra step at the start of the resolution process:

1. From the IP address, determine the IP pool.
2. From the IP address, determine the VR (mandatory constraint IpPool).
3. From the VR, determine the SAE that manages that VR.