Juniper
JN0-481 Exam

Data Center - Specialist

Questions & Answers
Demo

Version: 4.0

Question: 1

What is the primary reason for creating an Apstra worker node?

A. To support more than one blueprint
B. To create a space for storing event logs
C. To run Zero Touch Provisioning (ZTP)
D. To offload off-box agents and Intent-Based Analytics (IBA)

Answer: D

Explanation:

In Apstra 5.1, the worker node's primary purpose is to add scalable runtime capacity to an Apstra cluster by hosting off-box services that would otherwise consume resources on the controller. Specifically, worker nodes run containerized services such as off-box device agents (used to communicate with and manage devices) and Intent-Based Analytics (IBA) components (such as probes and analytics-related services). This design keeps the controller node focused on cluster management and control-plane functions (API handling, cluster-wide state, blueprint control workflows), while shifting resource-intensive operational services to worker nodes.

As your fabric grows—more switches, more telemetry, more devices requiring agent connectivity—CPU and memory demand increases notably, especially when IBA is enabled. Adding worker nodes allows you to scale those container workloads horizontally without redesigning the fabric or reducing analytics coverage. In a Juniper data center built on EVPN-VXLAN with Junos v24.4 leaf-spine roles, this separation helps ensure that Apstra can continuously validate intent, process streaming telemetry, and maintain device communications reliably at scale. Worker nodes therefore exist primarily to offload and scale operational agents and IBA services, improving performance and resilience for larger deployments.

Question: 2

Which Root Cause Identifier is currently supported in Juniper Apstra software?

A. Virtual network
B. Connectivity
C. ESI imbalance
D. BGP

Answer: B

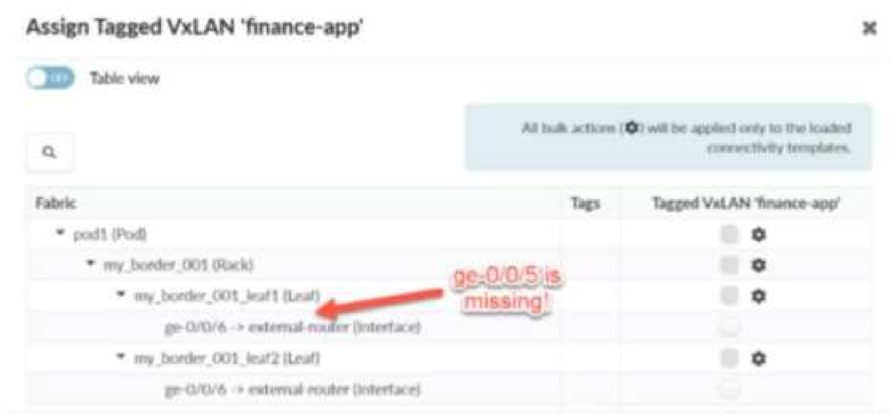
Explanation:

In Juniper Apstra 5.1, Root Cause Identification (RCI) is implemented with a currently supported model focused on connectivity. Practically, this means RCI is designed to take telemetry and state learned from the fabric (for example, interface operational status, LLDP neighbor information, and routing session status) and correlate those signals to determine the most likely underlying cause of a connectivity-impacting event. Within an EVPN-VXLAN IP fabric, many operational symptoms can appear similar at the service layer (endpoints cannot reach each other, routes disappear, overlays degrade), but RCI narrows the problem by correlating evidence across the underlay and control plane.

The “connectivity” RCI model targets common failure scenarios that directly break device-to-device reachability, such as a broken link, a miscabled link (wrong LLDP neighbors), or an operator-disabled interface. These conditions often cascade into higher-level symptoms, including BGP sessions dropping over affected links. With Junos v24.4-based leaf-spine fabrics, maintaining stable underlay connectivity is foundational for EVPN signaling and VXLAN forwarding; therefore, Apstra’s connectivity-focused RCI helps operators rapidly isolate whether the primary fault lies in physical adjacency, cabling/neighbor correctness, or administrative shutdown—reducing mean time to repair by pointing to the most probable root cause rather than only listing alarms.

Question: 3

You are attempting to attach the server connected to the my_border_001_leaf1 node's ge-0/0/5 interface to the finance-app virtual network.



Assign Tagged VxLAN 'finance-app'

Table view

All bulk actions (⚙) will be applied only to the loaded connectivity templates.

Fabric	Tags	Tagged VxLAN 'finance-app'
▾ pod1 (Pod)		☐ ⚙
▾ my_border_001 (Rack)		☐ ⚙
▾ my_border_001_leaf1 (Leaf)		☐ ⚙
ge-0/0/6 -> external-router (Interface)		☐
▾ my_border_001_leaf2 (Leaf)		☐ ⚙
ge-0/0/6 -> external-router (Interface)		☐

Referring to the exhibit, what would you do to solve the problem?

A. You can add a generic system to the physical topology.
B. You can set the generic system to the deploy mode.
C. You can allocate an IP pool resource to the virtual network.
D. You can assign the finance-app virtual network to the my_border_001_leaf1 node.

Answer: A

Explanation:

In Apstra 5.1, servers are modeled as Generic Systems and must be represented in the blueprint topology so that Apstra can bind an endpoint (the server) to a specific switch interface and then apply the intended connectivity template / virtual network attachment. In the exhibit, the interface ge-0/0/5 on my_border_001_leaf1 is shown as missing from the assignment workflow, which indicates that Apstra does not currently have an endpoint object connected to that port in the blueprint's staged physical topology (or that the port is not presented as an eligible connectivity point for server attachment).

The correct remediation is to add a Generic System connected to my_border_001_leaf1 ge-0/0/5 in Staged > Physical > Topology, thereby creating a modeled server link on that interface. Once the Generic System exists and the interface is a recognized server-facing connectivity point, you can assign the Tagged VXLAN “finance-app” connectivity template (or the VN assignment action driven by that template) to the server-facing interface and then commit the staged changes.

Changing “deploy mode” may affect whether Apstra actively configures a generic system-facing link, but it does not solve a missing interface in the topology model. Likewise, allocating an IP pool is unrelated to making the port available for attachment, and assigning the VN to the switch node is not how server interfaces are attached in this workflow.

Verified Juniper sources (URLs):

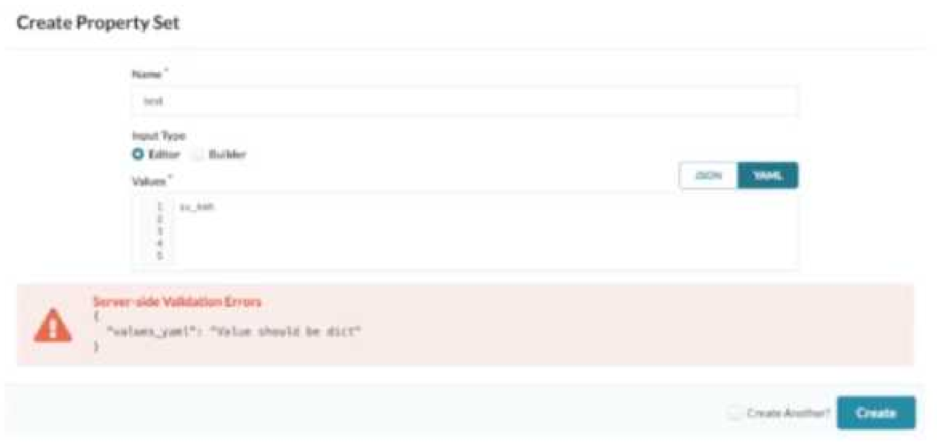
https://www.juniper.net/documentation/us/en/software/apstra5.1/apstra-user-guide/topics/topic-map/internal-generic-system-create.html

https://www.juniper.net/documentation/us/en/software/apstra5.0/apstra-user-guide/topics/topic-map/virtual-network-assignment-update.html

https://cloudlabs.apstra.com/labguide/Cloudlabs/6.0.0/test-drive-guide/lab1-junos-11_adding-gs.html

Question: 4

You are using Juniper Apstra to create your DC fabric. The fabric requires the use of configlets and requires a property set, which you call “test.” While creating the property set, you encounter an error message.


Create Property Set

Name *: test

Input Type: [✓ Editor] Builder

Values *: JSON [✓ YAML]

```
ss_kah
```

Server-side Validation Errors

```
{
  "values_yaml": "Value should be dict"
}
```

Create Another? Create

Referring to the exhibit, how would you correct the error?

A. Use the Builder option for input type of YAML.

B. Remove the trailing blank lines.

C. Change to JSON and click Create.

D. Use valid YAML syntax of key: value.

Answer: D

Explanation:

In Apstra 5.1, a property set is a structured data object used to parameterize configlets (config templates). The key point is that Apstra expects the property set “values” to be a dictionary/map so that the configlet can reference variables by name (for example, {{ NTP_SRV1 }} or nested keys). The exhibit shows a server-side validation error indicating that values_yaml “should be dict,” which occurs when the YAML content is entered as a single scalar string (such as try_ksh) instead of a key-value mapping.

To correct this, rewrite the YAML using valid key: value syntax so the top-level structure is a dictionary. For example, a minimal valid property set would look like role: try_ksh (or any meaningful key name aligned to the variables your configlet expects). If multiple variables are needed, add additional keys, and if your configlet uses nested objects, represent them as nested YAML dictionaries. This correction aligns the property set with Apstra’s intent-based model: values are stored as named properties and then rendered deterministically into device configuration. This is independent of Junos v24.4 specifics; Junos becomes relevant when the rendered configlet content is applied to devices, but the property set itself must first validate as a dictionary for Apstra to render the template correctly.

Verified Juniper sources (URLs):

https://www.juniper.net/documentation/us/en/software/apstra5.1/apstra-user-guide/topics/task/property-set-datacenter-design-create.html

https://www.juniper.net/documentation/us/en/software/apstra5.1/apstra-user-guide/topics/concept/property-set-datacenter-design.html

https://www.juniper.net/documentation/us/en/software/apstra5.1/apstra-user-guide/topics/ref/property-sets-api.html

Question: 5

You want to route between tenants in a multitenant environment in Juniper Apstr

a. What are two ways to accomplish this task? (Choose two.)

A. Route between VRFs on a VTEP-enabled device.

B. Use an external device to route between tenants.

C. Use iBGP to route within the same AS number.

D. Use virtual networks to route between VRFs.

Answer: A, B

Explanation:

In Apstra 5.1 multitenancy, tenants are modeled as routing zones, and each routing zone maps to a distinct VRF to provide strict Layer 3 isolation. Because each tenant’s VRF is separate, “routing between tenants” is effectively inter-VRF routing. Apstra’s routing-zone behavior emphasizes that inter-tenant routing is achieved via external systems: you connect each tenant/routing zone to an external router or firewall (often attached to border leafs), and that external device performs the policy-controlled inter-VRF routing between tenants. This approach is the most common because it centralizes security and compliance controls (stateful inspection, zone policies, NAT, logging) on the firewall/router while keeping the fabric clean and consistent.

A second method is to perform inter-VRF routing on a VTEP-capable border leaf that terminates the tenant VRFs. In EVPN-VXLAN designs, border leafs are frequently the demarcation where tenant VRFs connect to outside domains; when the same border leaf hosts multiple tenant VRFs and is designed to provide L3 services for them, it can act as the routing point between VRFs (subject to your design and security requirements). Junos v24.4 supports VRFs and policy constructs required for controlled route exchange and forwarding behavior, but Apstra's intent model still expects routing-zone isolation by default—so any inter-tenant connectivity should be explicitly designed and governed, typically at the border.