MCPA-LEVEL1 Q&As

MuleSoft Certified Platform Architect - Level 1

Pass Mulesoft MCPA-LEVEL1 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.pass4itsure.com/mulesoft-certified-platform-architect-level-1.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Mulesoft Official Exam Center

- **Instant Download** After Purchase
- **100% Money Back** Guarantee
- **365 Days** Free Update
- **800,000+** Satisfied Customers

QUESTION 1

An Anypoint Platform organization has been configured with an external identity provider (IdP) for identity management and client management. What credentials or token must be provided to Anypoint CLI to execute commands against the Anypoint Platform APIs?

A. The credentials provided by the IdP for identity management

B. The credentials provided by the IdP for client management

C. An OAuth 2.0 token generated using the credentials provided by the IdP for client management

D. An OAuth 2.0 token generated using the credentials provided by the IdP for identity management

Correct Answer: A

The credentials provided by the IdP for identity management ********** Reference: https://docs.mulesoft.com/runtime-manager/anypoint-platform- cli#authentication

>> There is no support for OAuth 2.0 tokens from client/identity providers to authenticate via Anypoint CLI. Only possible tokens are "bearer tokens" that too only generated using Anypoint Organization/Environment Client Id and Secret from https://anypoint.mulesoft.com/accounts/login. Not the client credentials of client provider. So, OAuth 2.0 is not possible. More over, the token is mainly for API Manager purposes and not associated with a user. You can NOT use it to call most APIs (for example Cloudhub and etc) as per this Mulesoft Knowledge article.

>> The other option allowed by Anypoint CLI is to use client credentials. It is possible to use client credentials of a client provider but requires setting up Connected Apps in client management but such details are not given in the scenario explained in the question.

>> So only option left is to use user credentials from identify provider

QUESTION 2

A company requires Mule applications deployed to CloudHub to be isolated between non- production and production environments. This is so Mule applications deployed to non- production environments can only access backend systems running in their customer- hosted non-production environment, and so Mule applications deployed to production environments can only access backend systems running in their customer-hosted production environment. How does MuleSoft recommend modifying Mule applications, configuring environments, or changing infrastructure to support this type of per- environment isolation between Mule applications and backend systems?

A. Modify properties of Mule applications deployed to the production Anypoint Platform environments to prevent access from non-production Mule applications

B. Configure firewall rules in the infrastructure inside each customer-hosted environment so that only IP addresses from the corresponding Anypoint Platform environments are allowed to communicate with corresponding backend systems

C. Create non-production and production environments in different Anypoint Platform business groups

D. Create separate Anypoint VPCs for non-production and production environments, then configure connections to the backend systems in the corresponding customer-hosted environments

Correct Answer: D

Create separate Anypoint VPCs for non-production and production environments, then configure connections to the backend systems in the corresponding customer-hosted environments. ********** >> Creating different Business Groups does NOT make any difference w.r.t accessing the non-prod and prod customer-hosted environments. Still they will be accessing from both Business Groups unless process network restrictions are put in place. >> We need to modify or couple the Mule Application Implementations with the environment. In fact, we should never implements application coupled with environments by binding them in the properties. Only basic things like endpoint URL etc should be bundled in properties but not environment level access restrictions. >> IP addresses on CloudHub are dynamic until unless a special static addresses are assigned. So it is not possible to setup firewall rules in customer-hosted infrastrcture. More over, even if static IP addresses are assigned, there could be 100s of applications running on cloudhub and setting up rules for all of them would be a hectic task, non-maintainable and definitely got a good practice. >> The best practice recommended by Mulesoft (In fact any cloud provider), is to have your Anypoint VPCs seperated for Prod and Non-Prod and perform the VPC peering or VPN tunneling for these Anypoint VPCs to respective Prod and Non-Prod customer-hosted environment networks. : https://docs.mulesoft.com/runtime-manager/virtual-private-cloud

QUESTION 3

Select the correct Owner-Layer combinations from below options

A. 1. App Developers owns and focuses on Experience Layer APIs

2.

Central IT owns and focuses on Process Layer APIs

3.

LOB IT owns and focuses on System Layer APIs

B. 1. Central IT owns and focuses on Experience Layer APIs

2.

LOB IT owns and focuses on Process Layer APIs

3.

App Developers owns and focuses on System Layer APIs

C. 1. App Developers owns and focuses on Experience Layer APIs

2.

LOB IT owns and focuses on Process Layer APIs

3.

Central IT owns and focuses on System Layer APIs

Correct Answer: C

1.

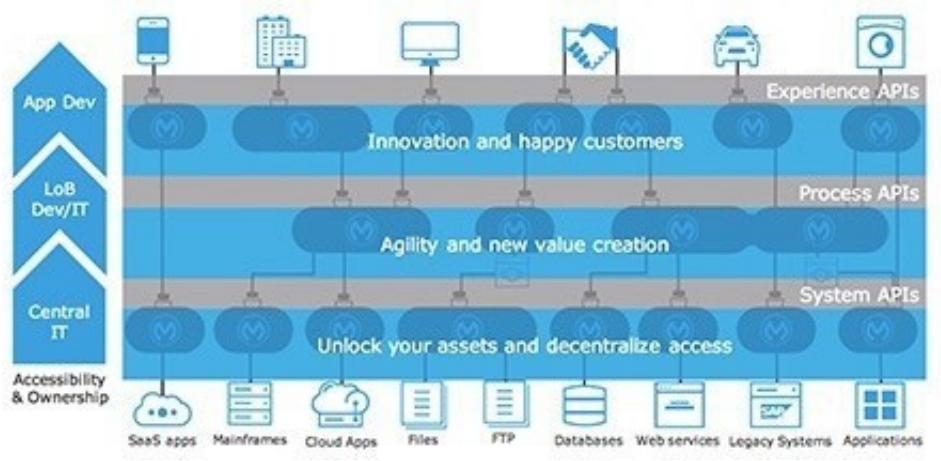
App Developers owns and focuses on Experience Layer APIs

2.

LOB IT owns and focuses on Process Layer APIs

3.

Central IT owns and focuses on System Layer APIs



References: https://blogs.mulesoft.com/biz/api/experience-api-ownership/ https://blogs.mulesoft.com/biz/api/process-api-ownership/ https://blogs.mulesoft.com/biz/api/system-api-ownership/

QUESTION 4

When designing an upstream API and its implementation, the development team has been advised to NOT set timeouts when invoking a downstream API, because that downstream API has no SLA that can be relied upon. This is the only downstream API dependency of that upstream API.

Assume the downstream API runs uninterrupted without crashing. What is the impact of this advice?

A. An SLA for the upstream API CANNOT be provided

B. The invocation of the downstream API will run to completion without timing out

C. A default timeout of 500 ms will automatically be applied by the Mule runtime in which the upstream API implementation executes

D. A toad-dependent timeout of less than 1000 ms will be applied by the Mule runtime in which the downstream API implementation executes

Correct Answer: A

An SLA for the upstream API CANNOT be provided.

****

>> First thing first, the default HTTP response timeout for HTTP connector is 10000 ms (10 seconds). NOT 500 ms.

>> Mule runtime does NOT apply any such "load-dependent" timeouts. There is no such behavior currently in Mule.

>> As there is default 10000 ms time out for HTTP connector, we CANNOT always guarantee that the invocation of the downstream API will run to completion without timing out due to its unreliable SLA times. If the response time crosses 10

seconds then the request may time out.

The main impact due to this is that a proper SLA for the upstream API CANNOT be provided.

Reference: https://docs.mulesoft.com/http-connector/1.5/http-documentation#parameters-3

QUESTION 5

A Mule application exposes an HTTPS endpoint and is deployed to the CloudHub Shared Worker Cloud. All traffic to that Mule application must stay inside the AWS VPC.

To what TCP port do API invocations to that Mule application need to be sent?

A. 443

B. 8081

C. 8091

D. 8082

Correct Answer: D

****

>> 8091 and 8092 ports are to be used when keeping your HTTP and HTTPS app private to the LOCAL VPC respectively.

>> Above TWO ports are not for Shared AWS VPC/ Shared Worker Cloud. >> 8081 is to be used when exposing your HTTP endpoint app to the internet through Shared LB >> 8082 is to be used when exposing your HTTPS endpoint app to

the internet through Shared LB

So, API invocations should be sent to port 8082 when calling this HTTPS based app.

References:

https://docs.mulesoft.com/runtime-manager/cloudhub-networking-guide

https://help.mulesoft.com/s/article/Configure-Cloudhub-Application-to-Send-a-HTTPS- Request-Directly-to-Another-Cloudhub-Application https://help.mulesoft.com/s/question/0D52T00004mXXULSA4/multiple-http-listerners-on-cloudhubone-with-port-9090