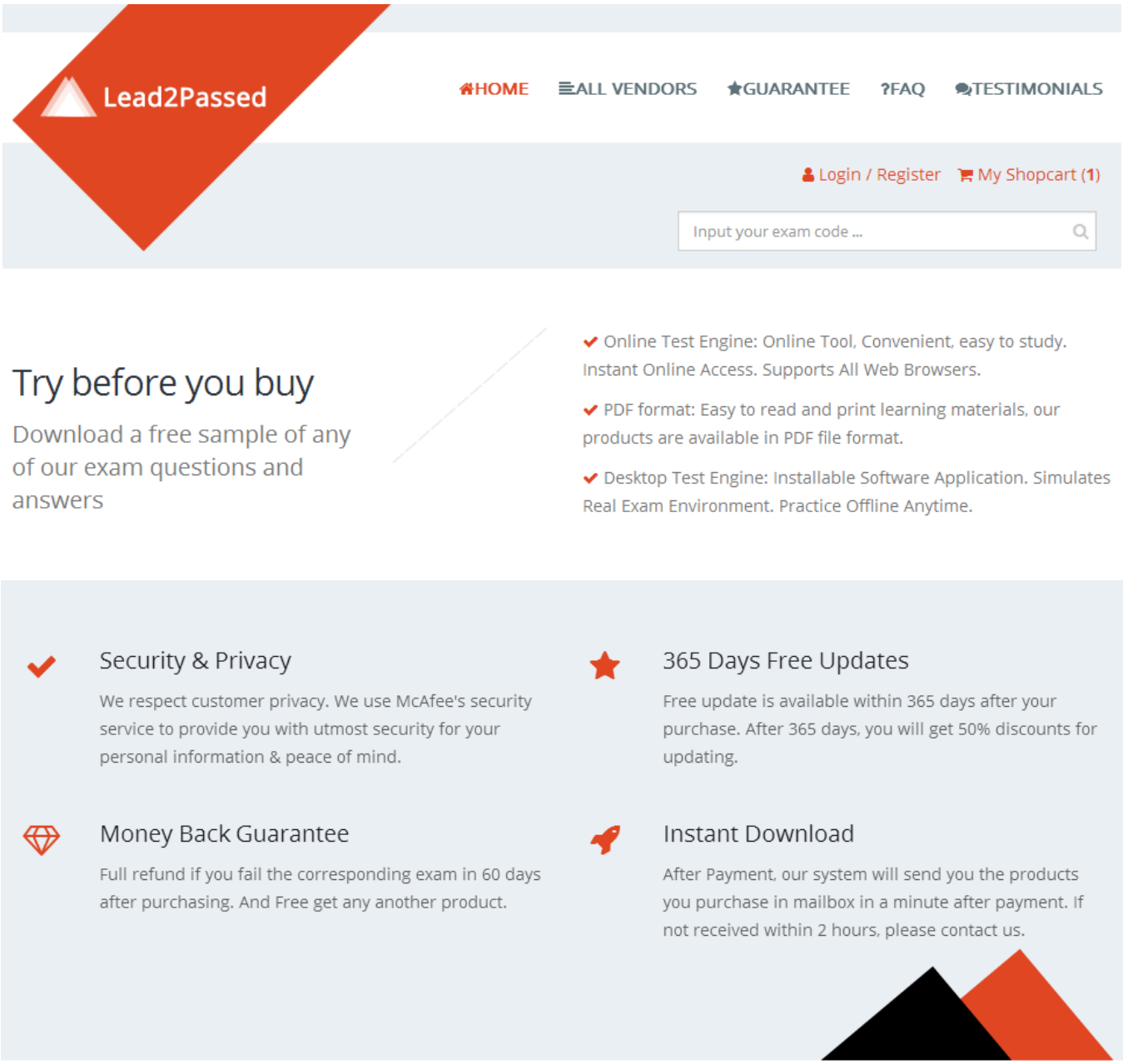
Lead2Passed



http://www.lead2passed.com

Valid Certification Exam Dumps Materials and Study Guide - Lead2Passed

Exam : **500-430**

Title : Cisco AppDynamics Professional Implementer

Vendor : Cisco

Version : DEMO

NO.1 Which URL retrieves all AppDynamics business transactions from an application using the AppDynamics Rest API?

A. http(s)://<controller-host>:<port=/controller/rest/applications/<application_name>/allbts

B. http(s)://<controller-host>:<port>/controller/rest/applications/<application_name=>/business-transactions

C. http(s)://<controller-host>:<port>/controller/applications/<application_name>/business-transactions

D. httpis)://<controller-host>:<port>/controller/applications/<application_name=>/allbis

Answer: B

Explanation

The AppDynamics Rest API to retrieve business transactions allows you to get a list of all business transactions in a business application, along with their key metrics and properties1. The correct URL format for this API is1:

http(s)

://<controller-host>:<port>/controller/rest/applications/<application_name>/business-transactions

The other options are incorrect because12:

Option A uses an invalid endpoint /allbts, which does not exist in the API.

Option C uses an incorrect base URL /controller/applications, which is used for the Controller UI, not the Rest API.

Option D uses a misspelled endpoint /allbis, which does not exist in the API. References: Retrieve All Business Transactions in a Business Application, AppDynamics APIs

NO.2 What are two valid reasons for using the REST API to retrieve health rule violations? (Choose two.)

A. For updating an AppDynamics dashboard

B. For determining which actions have been executed

C. When searching for historical events

D. For sending emails

E. When pushing events to the Event Management System is NOT possible

Answer: B C

Explanation

According to the Cisco AppDynamics Professional Implementer (CAPI) documents, the REST API for health rule violations allows you to retrieve information about the health rule violations that occurred in a specified time range for a given application1. You can use the REST API for health rule violations for the following valid reasons:

For determining which actions have been executed (B): The REST API response includes the details of the actions that were triggered by the health rule violation, such as email, SMS, HTTP request, or custom action1. You can use this information to verify if the actions were executed successfully, or to troubleshoot any issues with the action execution.

When searching for historical events : The REST API allows you to specify a custom time range for retrieving the health rule violations, such as BEFORE_TIME, AFTER_TIME, BETWEEN_TIMES, or BEFORE_NOW1. You can use this feature to search for historical events that occurred in the past, or to analyze the trends and patterns of the health rule violations over time.

The incorrect options are:

For updating an AppDynamics dashboard (A): This is not a valid reason for using the REST API for

health rule violations, because the AppDynamics dashboards already display the health rule violations that occurred in the selected time frame, along with the severity, status, affected entities, and actions2.
You do not need to use the REST API to update the dashboard, as the dashboard is automatically refreshed with the latest data from the Controller.
For sending emails (D): This is not a valid reason for using the REST API for health rule violations, because the REST API does not send emails directly. The REST API only returns the information about the health rule violations, and the actions that were triggered by them. If you want to send emails based on the health rule violations, you need to configure an email action in the health rule configuration, or use a custom action that invokes an external email service3.
When pushing events to the Event Management System is NOT possible (E): This is not a valid reason for using the REST API for health rule violations, because the REST API does not push events to the Event Management System. The REST API only returns the information about the health rule violations, and the actions that were triggered by them. If you want to push events to the Event Management System, you need to configure an HTTP request action in the health rule configuration, or use a custom action that invokes an external API3.
References:
1: Health Rule Violations API - AppDynamics
2: Health Rule Violations - AppDynamics
3: Actions - AppDynamics

NO.3 The AppDynamics Controller is instrumented by an internal, out-of-the-box, AppDynamics Java agent. Which account and user name are used to connectto the Controller to view theinformation provided by the internal AppDynamics agent?

A. The account is 'root' and the user is 'admin'.

B. The account is 'customer!' and the user is 'root'.

C. The account is 'system' and the user is "root.

D. The account is internal' and the user is 'admin'.

Answer: C

Explanation
The AppDynamics Controller is instrumented by an internal, out-of-the-box, AppDynamics Java agent that monitors the performance and health of the Controller itself1. To access the information provided by the internal agent, you need to log in to the Controller UI with the following credentials2 :
Account = system
Username = root
Password = <root_user_password>
The system account is a special account that is used only for internal monitoring and troubleshooting purposes. It is not visible in the normal Controller UI and requires a special URL to access it2. The root user is the default administrator user for the system account and has the same password as the admin user for the customer1 account3. References: Controller Self-Monitoring, Monitoring a Controller Using the Internal Monitoring Agent, Controller Accounts

NO.4 Which two statements are true about instrumenting standalone Windows services with NET Agent? (Choose two.)

A. AppDynamics .NET Agent does NOT support instrumenting process running multiple App domains.

B. AppDynamics.NET Agent automatically discovers all the Windows services to be instrumented.
C. AppDynamics.NET Agent can instrument both 32-bit as well 64-bit processes.
D. AppDynamics.NET Agent requires that the Windows services is running under the "App.pool identity user" account.
E. AppDynamics .NET Agent supports instrumentation of multiple instances of the same application.
Answer: C E
Explanation
The AppDynamics .NET Agent can instrument both 32-bit and 64-bit processes, as long as they are running on a supported .NET Framework version and operating system. The agent automatically detects the process architecture and loads the appropriate profiler DLL. You can also specify the process architecture manually in the agent configuration file1. The AppDynamics .NET Agent also supports instrumentation of multiple instances of the same application, such as Windows services or standalone applications. You can configure the agent to assign different tier and node names for each instance, based on the process name, process ID, or command line arguments. This allows you to monitor the performance and health of each instance separately2.
The other statements are false because:
A: AppDynamics .NET Agent does support instrumenting processes running multiple App domains. The agent can monitor multiple App domains within a single process, as long as they are running the same
.NET Framework version. The agent can also monitor multiple applications within a single App domain, by using the Standalone Applications element in the agent configuration file3.
B: AppDynamics .NET Agent does not automatically discover all the Windows services to be instrumented. The agent automatically instruments IIS applications only. For Windows services or standalone applications, you need to manually configure the agent by editing the agent configuration file and adding the Windows Services or Standalone Applications element. You also need to provide the executable name, tier name, and node name for each service or application4.
D: AppDynamics .NET Agent does not require that the Windows services are running under the "App.pool identity user" account. The agent can instrument Windows services running under any user account, as long as the account has sufficient permissions to load the agent profiler DLL and access the agent configuration and log files. The "App.pool identity user" account is only required for IIS applications that run in an application pool5.
References: .NET Agent Configuration Properties, Configure the .NET Agent for Windows Services and Standalone Applications, Instrument the DefaultDomain for Standalone Applications, Install the .NET Agent for Windows, Administer the .NET Agent

NO.5 What becomes more important as an AppDynamics Controller grows beyond supporting 500 agents?
A. CPU utilization
B. RAM allocated to the Controller
C. Network throughput
D. Disk VO
E. Thread count on the GlassFish server
Answer: C
Explanation
As an AppDynamics Controller grows beyond supporting 500 agents, network throughput becomes

more important. This is because the Controller needs to handle a large volume of data from the agents, as well as serve requests from the UI and API clients. Network throughput is the measure of how much data can be transferred over a network in a given time. A low network throughput can cause delays, errors, or timeouts in the communication between the Controller and the agents or clients. Therefore, it is recommended to monitor the network throughput of the Controller and ensure that it meets the minimum requirements for the expected load123. References: Controller System Requirements, Performance and Controller Sizing Guidelines, How to Run AppDynamics in Microsoft Azure