DUMPSARENA

Palo Alto Networks XDR Engineer

Palo Alto Networks XDR-Engineer

Version Demo

Total Demo Questions: 7

Total Premium Questions: 70

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

Topic Break Down

Topic	No. of Questions
Topic 1, Architecture and Design	2
Topic 2, Deployment and Configuration	29
Topic 3, Data Collection	15
Topic 4, Detection and Investigation	17
Topic 5, Response and Automation	7
Total	70

QUESTION NO: 1

An engineer is preparing Pathfinder discovery by using Kerberos authentication. Discovery targets are in several internal subnets, and the engineer wants to prevent name-resolution issues from causing authentication failures. Which two DNS settings should be validated? (Choose two.)

A. DNS forwarders

B. Reverse DNS zones

C. Reverse DNS records

D. AD DS-integrated zones

ANSWER: B C

Explanation:

Reverse DNS zones and accurate **reverse DNS records** are required considerations for Kerberos-based Pathfinder discovery. The reverse zone supports reverse lookups for the scanned address ranges, and PTR records map discovered IP addresses to the corresponding host names.

DNS forwarders can help resolve external or otherwise delegated names, but they do not establish reverse mappings for the discovered hosts. An AD DS-integrated zone can be used in an Active Directory environment, but it is not itself the required reverse-lookup configuration.

QUESTION NO: 2

When using Kerberos as the authentication method for Pathfinder, which two settings must be validated on the DNS server? (Choose two.)

A. DNS forwarders

B. Reverse DNS zone

C. Reverse DNS records

D. AD DS-integrated zones

ANSWER: B C

Explanation:

Reverse DNS zone and **Reverse DNS records** must be validated when Pathfinder uses Kerberos authentication. A reverse DNS zone provides the DNS namespace used for reverse lookups, while reverse DNS records—PTR records within that zone—map endpoint IP addresses back to their fully qualified host names. Together, these DNS components allow Pathfinder to reliably identify discovered devices and associate their network addresses with the host identities used during authenticated discovery. Accurate IP-to-hostname resolution is especially important in an Active Directory and Kerberos environment, where service and host identity resolution must be consistent with the domain's DNS configuration. Before deploying or troubleshooting Kerberos-based Pathfinder scans, administrators should therefore confirm that the applicable reverse lookup zone exists, that it covers the scanned network ranges, and that relevant systems have correct PTR records pointing to their canonical names. Palo Alto Networks documents Pathfinder as part of Cortex XDR's unmanaged-device discovery capabilities; its configuration should be planned alongside the organization's authentication and name-resolution infrastructure. For background on Pathfinder, see the Cortex XDR Pathfinder documentation. Microsoft's DNS zone management documentation also describes reverse lookup zones and the records they contain.

QUESTION NO: 3

Which action is being taken with the query below?

dataset = xdr_data

| fields agent_hostname, _time, _product

| comp latest as latest_time by agent_hostname, _product

| join type=inner (dataset = endpoints

| fields endpoint_name, endpoint_status, endpoint_type) as lookup lookup.endpoint_name = agent_hostname

| filter endpoint_status = ENUM.CONNECTED

| fields agent_hostname, endpoint_status, latest_time, _product

A. Monitoring the latest activity of endpoints

B. Identifying endpoints that have disconnected from the network

C. Monitoring the latest activity of connected firewall endpoints

D. Checking for endpoints with outdated agent versions

ANSWER: A

Explanation:

Monitoring the latest activity of endpoints is correct because the query starts with event records in the `xdr_data` dataset and retains each event's hostname, timestamp, and product. The `comp latest as latest_time by agent_hostname, _product` stage aggregates those records to produce the most recent observed timestamp for every hostname-and-product combination. This is the core activity-monitoring action: it converts potentially many events into a current last-seen value for each endpoint source.

The query then enriches that activity data by performing an inner join with the `endpoints` dataset, matching `endpoint_name` to `agent_hostname`. Endpoint metadata, including its connection status, is therefore associated with the calculated latest activity time. Filtering on `endpoint_status = ENUM.CONNECTED` limits the resulting view to endpoints currently reported as connected. The final projection displays the hostname, connection status, most recent activity time, and product, making the output suitable for reviewing recent activity among active endpoints.

This use of dataset selection, aggregation, joins, filtering, and field projection follows Cortex XDR XQL query workflows documented in the Cortex XDR documentation and the XQL Language Reference.

QUESTION NO: 4

A regulated business unit requires a controlled Cortex XDR agent rollout. The team wants to avoid unplanned feature-level agent changes while still receiving updated protection content after a validation period. Which two agent settings should be configured? (Choose two.)

A. Set the agent upgrade scope to maintenance releases only

B. Enable content auto-update with a four-day delay

C. Disable content auto-update for the regulated endpoints

D. Enable minor content version updates without a delay

ANSWER: A B

Explanation:

Limiting the **agent upgrade scope to maintenance releases only** helps the business unit avoid broader feature-level changes that may require extensive compatibility testing and formal change approval.

Enabling **content auto-update with a four-day delay** allows prevention and detection content to be updated automatically after the organization has had time to validate the release in less sensitive environments. Disabling content updates entirely would unnecessarily reduce protection, while minor-content settings do not provide the requested controlled delay.

QUESTION NO: 5

An engineer is building a dashboard to visualize the number of alerts from various sources. One of the widgets from the dashboard is shown in the image below:

The engineer wants to configure a drilldown on this widget to allow dashboard users to select any of the alert names and view those alerts with additional relevant details. The engineer has configured the following XQL query to meet the requirement:

dataset = alerts

| fields alert_name, description, alert_source, severity, original_tags, alert_id, incident_id

| filter alert_name =

| sort desc _time

How will the engineer complete the third line of the query (filter alert_name =) to allow dynamic filtering on a selected alert name?

A. $y_axis.value

B. $x_axis.value

C. $x_axis.name

D. $y_axis.name

ANSWER: B

Explanation:

`$x_axis.value` is the appropriate drilldown variable because the dashboard visualization places the alert-name category on the x-axis. When a user selects a bar, point, or other plotted item representing an alert name, Cortex XDR resolves this variable to the value of that selected x-axis category. The resulting filter is therefore evaluated with the selected alert name, returning alert records whose `alert_name` matches the dashboard selection. The remaining query fields then provide the requested contextual details, including description, source, severity, tags, alert ID, and incident ID, while descending time sorting presents the most recent matching alerts first.

Drilldowns are designed to pass the selected visualization context into an XQL query rather than requiring a static alert-name literal. Using the x-axis value preserves that dynamic behavior for every alert-name category displayed in the widget. This is especially useful in an alert-count visualization, where the plotted numeric measure represents the count, but the category value identifies the alert name that must be used as the filter criterion. For dashboard and XQL guidance, see the Palo Alto Networks Cortex documentation portal and the Cortex XDR XQL Language Reference.

QUESTION NO: 6

Which components may be included in a Cortex XDR content update?

A. Device control profiles, agent versions, and kernel support

B. Behavioral Threat Protection (BTP) rules and local analysis logic

C. Antivirus definitions and agent versions

D. Firewall rules and antivirus definitions

ANSWER: B

Explanation:

Behavioral Threat Protection (BTP) rules and local analysis logic is correct because Cortex XDR content updates deliver detection-related content independently of a Cortex XDR agent software-version upgrade. Behavioral Threat Protection rules

provide endpoint behavioral detection content that the agent uses to identify suspicious activity and known malicious techniques. Local analysis logic provides updated endpoint-side analytical capabilities, allowing the agent to evaluate activity and files using the latest available detection logic. This update mechanism enables Palo Alto Networks to improve prevention and detection coverage promptly, without requiring administrators to deploy a new agent package solely to obtain the latest security content. Administrators can view and manage content versions separately from agent versions in Cortex XDR, which is important when validating that endpoints have received current protection content. Palo Alto Networks documents content updates as a distinct update type and explains the role of Behavioral Threat Protection in endpoint prevention. See the Cortex XDR documentation on managing content updates and the Behavioral Threat Protection documentation.

QUESTION NO: 7

Based on the Malware profile image below, what happens when a new custom-developed application attempts to execute on an endpoint?

A. It will immediately execute

B. It will not execute

C. It will execute after one hour

D. It will execute after the second attempt

ANSWER: B

Explanation:

It will not execute is correct when the Malware profile is configured to block unknown executable files pending a WildFire verdict, as indicated by the intended profile-based scenario. A newly custom-developed application generally has no existing WildFire reputation or verdict when it is first encountered. With unknown-file execution blocking enabled, the Cortex XDR agent prevents the executable from starting rather than allowing the process to run while its reputation is unknown. This provides pre-execution protection against a newly introduced executable that could potentially be malicious.

The agent can submit eligible unknown files for WildFire analysis and apply the resulting verdict when it becomes available. If the file is subsequently determined to be benign, or an administrator creates an appropriate allow rule or exclusion, execution can be permitted according to the configured policy. The important behavior for the initial attempt is that the unknown application is stopped by the endpoint protection policy; it is not governed by a fixed one-hour wait or an attempt-count threshold. Palo Alto Networks documents Malware Protection profiles and their WildFire-based prevention capabilities in the Cortex XDR documentation portal. For background on cloud-based analysis and verdicts used by endpoint protections, see Palo Alto Networks WildFire.