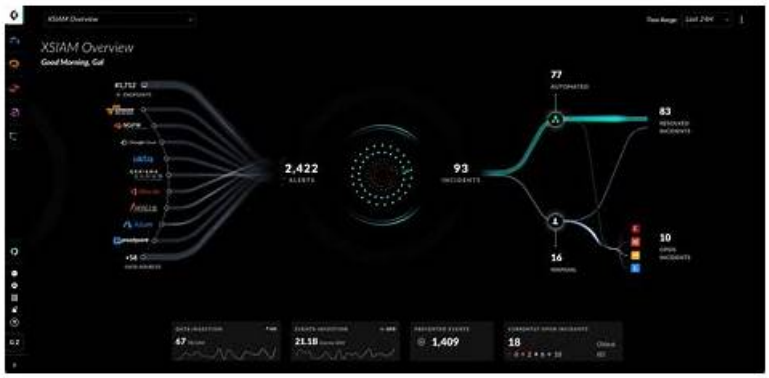
更新するXSIAM-Engineer｜一番優秀なXSIAM-Engineer資格認証攻略試験｜試験の準備方法Palo Alto Networks XSIAM Engineer更新版



ちなみに、ShikenPASS XSIAM-Engineerの一部をクラウドストレージからダウンロードできます：https://drive.google.com/open?id=1ys0qhS8tjnrJlSmmk7OwXdq26TIC8RlY

ShikenPASSガイドトレントは、専門家によって編集され、経験豊富な専門家によって承認されています。言語は理解しやすいため、どの学習者にも学習上の障害はなく、XSIAM-Engineer学習質問はどの学習者にも適しています。このソフトウェアは、さまざまな自己学習および自己評価機能を強化して、学習の結果を確認します。このソフトウェアは、学習者が脆弱なリンクを見つけて対処するのに役立ちます。 XSIAM-Engineer試験トレントは、タイミング機能と試験を刺激する機能を向上させます。 Palo Alto Networks XSIAM Engineerラーニングガイドを使用すると、XSIAM-Engineer試験に簡単に合格できます。

Palo Alto Networks XSIAM-Engineer 認定試験の出題範囲：

トピック	出題範囲
トピック 1	• メンテナンスとトラブルシューティング：このセクションでは、セキュリティ運用エンジニアのスキルを評価し、XSIAMコンポーネントの導入後のメンテナンスとトラブルシューティングを網羅します。例外設定の管理、XDRエージェントやBroker VMなどのソフトウェアコンポーネントの更新、データの取り込み、正規化、解析に関する問題の診断などが含まれます。受験者は、運用の信頼性を確保するために、統合、自動化プレイブック、システムパフォーマンスのトラブルシューティングも実施する必要があります。
トピック 2	• 計画とインストール：このセクションでは、XSIAMエンジニアのスキルを評価し、Palo Alto Networks Cortex XSIAMコンポーネントの計画、評価、インストールについて学習します。既存のITインフラストラクチャの評価、ハードウェア、ソフトウェア、および統合に関する導入要件の定義、そしてXSIAMアーキテクチャの通信ニーズの確立に重点を置いています。受験者は、エージェント、ブローカーVM、エンジンの設定に加え、ユーザーロール、権限、アクセス制御の管理も行う必要があります。
トピック 3	• 統合と自動化：この試験セクションでは、SIEMエンジニアのスキルを評価し、XSIAMにおけるデータのオンボーディングと自動化の設定に焦点を当てます。エンドポイント、ネットワーク、クラウド、IDなどの多様なデータソースの統合、メッセージング、認証、脅威インテリジェンスなどの自動化フィードの設定、マーケットプレイスコンテンツパックの実装などを網羅します。また、効率的なワークフロー自動化のためのプレイブックの計画、作成、カスタマイズ、デバッグ能力も評価されます。

トピック 4	• コンテンツ最適化：この試験セクションでは、検知エンジニアのスキルを評価し、XSIAMコンテンツと検知ロジックの改良に焦点を当てます。正規化のための解析およびデータモデリングルールの導入、相関関係、IOC、BIOC、攻撃対象領域管理に基づく検知ルールの管理、インシデントおよびアラートレイアウトの最適化などが含まれます。受験者は、運用の可視性を高めるためのカスタムダッシュボードとレポートテンプレートの作成能力も証明する必要があります。

>> XSIAM-Engineer資格認証攻略 <<

素敵なXSIAM-Engineer資格認証攻略試験-試験の準備方法-100％合格率のXSIAM-Engineer更新版

概念、質問の種類、デザイナーのトレーニングなどの状況改革に応じて当社。最新のXSIAM-Engineer試験トレントは、多くの専門家や教授によって設計されました。 XSIAM-Engineerクイズ準備を使用する場合は、デモについて学ぶ機会があります。さまざまなテキストタイプと、デモでそれらにアプローチする最善の方法を認識することは非常に重要です。同時に、当社のXSIAM-Engineerクイズトレントは、お客様がXSIAM-Engineer試験に合格するのを助けるために、クローズテストの機能とルールをまとめました。

Palo Alto Networks XSIAM Engineer 認定 XSIAM-Engineer 試験問題 (Q102-Q107):

質問 # 102
During a new Cortex XSIAM deployment, a user consistently experiences timeout sessions while trying to connect to the agent through Live Terminal, even though the firewall engineer has confirmed that all source IP addresses, port 443, and destinations are allowed.
What could be causing these persistent timeout issues?

- A. User does not have administrative privileges on the managed endpoint.
- B. Live Terminal feature is not supported on the current OS.
- C. SSL Decryption is currently being used to inspect the underlying traffic.
- D. NTP is not synchronized with the server time.

正解：C

解説：
Persistent timeout issues with Cortex XSIAM Live Terminal, despite firewall rules being open, are often caused by SSL Decryption inspecting the traffic. Live Terminal relies on secure, end-to-end TLS communication, and decryption breaks this channel, leading to session failures.

質問 # 103
You are responsible for a large XSIAM deployment with Broker VMS deployed across multiple on-premises data centers, behind firewalls and proxies. You receive a critical security bulletin from Palo Alto Networks regarding a vulnerability in a specific Broker VM firmware version, requiring an immediate update to version 2.1.3. However, your internal change management policy mandates a maximum 2-day outage window for all non-critical updates. You need to identify the potential bottlenecks and a strategy to minimize downtime while ensuring the update's success. Which of the following considerations and actions are crucial for a successful, low- downtime Broker VM firmware update in this scenario? (Select all that apply)

- A. Temporarily disable all XDR Agents reporting to the Broker VMS to prevent data loss during the update process and re-enable them after successful completion.
- B. Back up the Broker VM configuration and take a snapshot of the virtual machine before initiating the firmware update to facilitate quick recovery in case of an unforeseen issue.
- C. Verify network connectivity and firewall rules from each Broker VM to the XSIAM cloud update servers before initiating the update, specifically checking for newly introduced FQDNs or ports in the 2.1.3 release notes.
- D. Pre-download the Broker VM firmware image to a local, accessible server within each data center to bypass potential internet bandwidth or proxy issues during the update.
- E. Ensure that redundant Broker VMS are deployed in each data center and update them sequentially (e.g., one at a time) to

maintain continuous data ingestion and minimize service disruption.

正解：B、C、D、E

解説：
This question tests a comprehensive understanding of managing critical updates in complex environments. A: Pre-downloading firmware is crucial for large deployments behind proxies/firewalls, as it eliminates potential network delays or failures during the critical update window, ensuring the update package is readily available. B: Verifying network connectivity and firewall rules is paramount. Firmware updates can sometimes introduce new communication requirements, and pre-checking FQDNs/ports prevents 'update failed' issues due to unexpected network blocks. C: Redundant Broker VMS and sequential updates are fundamental for minimizing downtime. Updating one VM at a time allows the other(s) to continue processing, ensuring continuous data ingestion. This directly addresses the 'low-downtime' requirement. D: Backing up configuration and snapshots provides a critical rollback mechanism. If an update fails catastrophically, restoring from a snapshot is often the fastest recovery path, minimizing the impact of unforeseen issues. E: Temporarily disabling XDR Agents is incorrect. This would cause significant data loss as agents would stop reporting. The goal is to minimize disruption, not cause it. Redundant Broker VMS (C) address continuous data ingestion during updates.

質問 # 104
A complex XSOAR playbook integrating with multiple external security tools (EDR, Firewall, IAM) is failing intermittently with a generic 'NoneType' object has no attribute 'get'' error in a Python script task. The script processes data returned from a previous EDR query command. You've confirmed the EDR query command sometimes returns valid data and sometimes returns 'null' or an empty list. The script snippet causing the error is as follows:

```
alert_details = demisto.get(demisto.incidents()[0], 'details') # Line X
if alert_details:
    host_name = alert_details.get('host_info', {}).get('hostname') # Line Y
    # Further processing...
else:
    demisto.log('No alert details found.')
```

Which of the following approaches will most effectively debug and resolve this issue while making the playbook more robust?

- A. Analyze the EDR query command's output for cases where it returns 'null' or an empty list, and modify the playbook logic to proactively handle these specific outputs before passing them to the script.
- B. Before Line X, add a check 'if demisto.incidents() and len(demisto.incidents()) > 0:' to ensure an incident object exists, and handle the case where it doesn't.
- C. Implement an explicit 'try-except AttributeError' block around Line Y to catch the 'NoneType' error and log the state of 'alert_details'.
- D. Ensure that the 'details' field in the incident context is always populated by an earlier playbook task, potentially using a 'Set' command with a default empty dictionary.
- E. Modify Line Y to 'host_name = alert_details and alert_details.get('host_info', to use short-circuiting for NoneType checks.

正解：A

解説：
The error 'NoneType' object has no attribute 'get'' at Line Y implies 'alert_details' is 'None'. The current 'if alert_details:' check should handle this if becomes *None' at that point. The problem is likely that 'details')' (Line X) itself is returning 'None' due to the EDR query's intermittent 'null' or empty list output. Option D directly addresses the root cause: the inconsistent output from the EDR query. By proactively handling these 'no data' scenarios before the script, the playbook becomes robust. Options A and B address potential 'NoneType' issues but don't solve the underlying data inconsistency. Option C is a reactive error handling, not a proactive solution. Option E attempts to force a default, but the EDR output itself needs robust handling.

質問 # 105
An XSIAM engineer discovers that a large number of 'Alert' events are being generated with duplicate or near-duplicate 'description' fields, making it difficult for analysts to triage effectively. For example, 'Suspicious login from new country' and 'Suspicious login from previously unseen country' are considered duplicates for practical purposes. To optimize content by normalizing these descriptions and potentially reducing alert fatigue, which combination of XSIAM data modeling rules and techniques would be most effective and resilient?

- A. Utilize XSIAM's 'Content Enrichment' framework to create a Python script that employs Natural Language Processing (NLP) techniques (e.g., stemming, lemmatization, semantic similarity algorithms) to generate a 'canonical_description' and store it. Then, use this new field for alert aggregation.

- B. Leverage XSIAM's 'Anomaly Detection Engine' to identify patterns in the 'description' field. Train a custom model to cluster similar descriptions together and then define an 'alert promotion rule' that only promotes one alert per cluster to the analyst queue.
- C. Manually create a comprehensive 'lookup table' mapping all known duplicate 'description' variants to a single 'master_description'. Deploy an 'ingestion mapping rule' to transform the 'description' field using this lookup table. For remaining variations, create a 'post-ingestion aggregation rule' that groups alerts by a 'hash' of the transformed description.
- D. Configure an 'XSIAM playbook' to automatically close duplicate alerts based on string similarity of their 'description' field every hour. For the remaining alerts, an 'alert grouping rule' should be set up to group alerts with identical 'description' values.
- E. Implement a 'regex extraction rule' on the 'description' field to capture key phrases and use these phrases to generate a 'normalized_alert_type' field. Subsequently, configure 'alert deduplication rules' based on this 'normalized_alert_type' and a defined time window.

正解：C、E

解説：
This question seeks a resilient and effective method to normalize near-duplicate alert descriptions and reduce fatigue. Option A is the most practical, scalable, and resilient approach within typical XSIAM content optimization capabilities: 1. Regex Extraction Rule : This is a core content optimization capability. Using regex to capture key phrases (Suspicious login', 'new country') from variable descriptions allows for a programmatic way to derive a 'normalized_alert_type' field. This field becomes a consistent, structured representation of the alert's core meaning, even if the raw description varies slightly. 2. Alert Deduplication Rules : XSIAM has built-in alert deduplication capabilities. By applying these rules on the newly created 'normalized_alert_type' field (along with other contextual fields like 'username', 'source_ip' , and a time window), you can effectively prevent multiple alerts with functionally identical meanings from reaching the analyst, reducing fatigue. This is a standard and robust method. Why other options are less optimal or practical: - B (NLP via Python script) : While semantically powerful, integrating custom NLP Python scripts for every incoming alert description at scale can be computationally expensive and difficult to maintain within the high-performance ingestion pipeline required by XSIAM. It's often overkill for common variations and might introduce latency. - C (Manual Lookup Table + Hashing) : Manually creating a comprehensive lookup table for all possible near-duplicates is not resilient or scalable. New variations would require constant manual updates. Hashing exact matches doesn't solve 'near-duplicate' problems. - D (Playbook to close duplicates) : This is a post-generation remediation step, not a content optimization step that normalizes the data itself to prevent the initial duplicates. Relying on playbooks to 'close' duplicates after they've been generated still means they've consumed resources and potentially caused initial noise. - E (Anomaly Detection Engine for Clustering) : While XSIAM has anomaly detection, using it for clustering alert descriptions specifically to then promote only one is not its primary design. Training and maintaining such a model for evolving text descriptions can be complex and resource-intensive, and the solution might be too abstract for the specific problem of 'near-duplicate descriptions'.

質問 # 106
An internal audit identified a gap in detecting privilege escalation attempts using Windows built-in tools like 'seclogon.exe' (RunAs) or psexec.exe' (Sysinternals) when used by non-administrative users. These tools are legitimate but often abused. The goal is to detect Process.Name' 'seclogon.exe' or 'psexec.exe' being invoked from a standard user context, especially when followed by an attempt to execute a sensitive command on another system or elevate privileges locally. Which XQL query would effectively capture this behavior as a BIOC, minimizing false positives from legitimate IT operations?

- A.
```
dataset = xdr_data | pattern (Process.Name in ('seclogon.exe', 'psexec.exe') and User.IsAdmin = false) as stage_1, (Process.CommandLine contains
('net localgroup Administrators', 'cmd.exe /c whoami /priv', 'powershell.exe -exec Bypass') and Process.ParentProcess.PID = stage_1.Process.PID)
as stage_2 within 10s by Host.ID, User.ID | where stage_1.Process.Reputation != 'trusted' and stage_2.Process.Reputation != 'trusted'
```
- B.
```
dataset = xdr_data | filter Process.Name = 'psexec.exe' AND Network.Connection == 'true'
```
- C.
```

```
- D.
```

```
- E.
```
dataset = xdr_data | filter Process.Name = 'seclogon.exe' AND Process.CommandLine contains 'runas'
```

正解：A

解説：
Option B is the most effective and precise XQL query. Option A is too broad and will generate many false positives from legitimate use of these tools by non-admin users for non-privileged tasks. Option C is too generic for psexec and misses seclogon. Option D is specific but misses other malicious uses. Option E is very broad and will generate many false positives. Option B accurately uses the

'pattern' command to look for the specific sequence: 'seclogon.exe' or 'psexec.exe' being invoked by a non-admin user (stage 1), immediately followed (within 10 seconds, and from the same host/user) by attempts to execute privilege-escalation-related commands (stage 2). The 'where stage_1-Process.Reputation != 'trusted' and stage_2.Process.Reputation != 'trusted'' further refines the detection by excluding known good executables, significantly reducing false positives while catching the intended behavior.

質問 # 107
.....

世界で、多くの人はXSIAM-Engineer学習教材を利用しています。ここから見ると、XSIAM-Engineer学習教材はいい資料です。彼らはXSIAM-Engineer学習教材を勉強したら、XSIAM-Engineer試験に合格しました。だから、彼らはXSIAM-Engineer学習教材に対して、感謝の気持ちです。つまり、あなたもXSIAM-Engineer学習教材を購入すれば、後悔することはありません。