zscaler™ Certification

STUDY GUIDE

Zscaler Digital Experience Administrator (ZDXA)

Certification Exam

.

Zscaler Digital Experience..... 6
How to Use This Study Guide..... 6
About the ZDXA Exam..... 6
Exam Format..... 6
Audience & Qualifications..... 7
Skills Required..... 7
Recommended Training..... 7
Using the Reference Architecture to Frame Your Study..... 9
Monitoring and Triaging Digital Experience (30%)..... 10
Understanding the ZDX Scoring Model..... 11
Lowest Performing Metric Logic..... 14
Role of Web Probes in Scoring..... 15
Role of Cloud Path Probes in Scoring..... 16
Interpreting Application vs User vs Device Scores..... 18
Using the Performance Dashboard for Triage..... 22
Performance Dashboard Overview..... 22
Identifying Incident Scope (User, Site, Region, Global)..... 23
Filtering by Application, Region, and User Groups..... 24
Identifying Impacted Users and Locations..... 26
Escalation Decision Framework..... 27
Zscaler Client Connector (ZCC) and ZDX Cloud Interaction..... 30
ZCC Telemetry Collection..... 30
Data Flow from Endpoint to ZDX Cloud..... 32
Dependencies on ZCC Health and Configuration..... 33
Application View Analysis..... 37
Comparing Application Performance..... 37
Identifying Degraded Applications..... 42
SaaS vs Private Application Monitoring..... 44
Device Health and Endpoint Telemetry..... 48
Device Scoring Dashboard..... 49
CPU and Memory Utilization Analysis..... 50
Wi-Fi and Local Connectivity Metrics..... 52
Identifying Endpoint-Induced Performance Issues..... 53
Using ZDX Copilot for Baseline Comparison..... 56
Copilot Overview..... 56
Comparing Current vs Historical Performance..... 57
Using Natural Language Queries for Analysis..... 59
Validating Copilot Findings..... 60

Third-Party Integrations and Ecosystem....64
Call Quality Monitoring (Teams, Zoom)....64
ServiceNow Integration....66
Microsoft Intune Integration....68
OneAPI and Programmatic Integration Use Cases....69
Network Intelligence and ISP Analysis....72
Last Mile Performance Analysis....72
Intermediate ISP Behavior....74
Identifying Network Anomalies....75
Differentiating Network vs Application Issues....76
ZDX API Fundamentals....80
API Overview and Capabilities....80
Use Cases for Automation and Reporting....83
Integrating ZDX Data into External Systems....84
Operational Configuration and Optimization (30%)....88
Configuring Applications and Monitoring Models....88
Predefined Applications....89
Custom Applications....90
Managed Monitoring and Hosted Monitoring....93
Real User Monitoring (RUM)....94
Selecting the Right Monitoring Approach....95
Configuring Probes....99
Web Probes....100
Cloud Path Probes....103
Hosted Probes....106
When to Use Each Probe Type....108
Probe Configuration Best Practices....109
Remediation and Optimization Scripts....114
Overview of ZDX Remediation Scripts....114
Uploading and Managing ScriptsExecuting Scripts at Scale....116
Prerequisites and Permissions....117
Safety and Change Control Considerations....118
Validating Remediation Outcomes....119
Browser-Based Application Performance Monitoring....122
Monitoring Real-Time User Experience....122
Browser-Based Performance Metrics....123
Business-Critical Application Monitoring....125
Validating End-User Experience....126
Troubleshooting and Root Cause Analysis (30%)....130
Analyzing Device, Network, and Application Telemetry....130
Endpoint Telemetry Analysis....130

Network Telemetry Analysis....131
Application Performance Indicators....133
Identifying Hardware vs Network vs Application Issues....134
Device Inventory and Event Correlation....135
Diagnostic Sessions and Log Analysis....137
Initiating Diagnostic Sessions....138
Interpreting Diagnostic Output....140
Reviewing Log Files....141
Determining Root Cause....142
Implementing and Validating Remediation....143
Network Path Analysis and Bottleneck Identification....146
End-to-End Path Visibility....146
Identifying Latency Sources....147
Local Network vs ISP vs Application Bottlenecks....149
Interpreting MTR-Style Path Data....150
Using Zscaler AI for Root Cause Analysis....153
Copilot for Troubleshooting....154
Deep Trace Analysis with AI....155
Identifying Process-Level Issues....156
Validating AI-Generated Insights....157
Reporting and Data Analysis....161
QBR and Trend Analysis....167
Organization-Wide Performance Insights....167
Snapshots and Shareable Views....171
Creating Snapshots....171
Sharing UI State....173
Using Shareable URLs for Collaboration....174
Organization-Wide Incident Management....177
Identifying Widespread Incidents....177
Wi-Fi Incident Analysis....179
Self-Service and Helpdesk Workflows....180
Incident Categorization and Routing....183
Administration and Operational Governance (10%)....186
Role-Based Access Control (RBAC)....186
RBAC Fundamentals....188
Service Desk vs Admin Roles....189
Super Admin Responsibilities....190
Least Privilege Access Design....192
Alerting and Notification Configuration....196
Alert Rule Design....197
Thresholds and Time Windows....199

AND/OR Logic in Alerts..... 200
Notification Methods (Email, Webhooks, ITSM)..... 201
ServiceNow and Webhook Integration.....202
OAuth 2.0 Basics for Alerting Integrations.....204
Admin Authentication and Security.....207
Admin Authentication Settings.....207
ZCC Profile Configuration.....209
Identity Server Settings.....210
Securing Administrative Access.....211

Zscaler Digital Experience Administrator

How to Use This Study Guide

Welcome to the Zscaler ZDXA Study Guide, which will serve as your go-to resource in preparing for the ZDXA exam and receiving your ZDXA certification.

About the ZDXA Exam

The Zscaler Digital Experience Administrator (ZDXA) is a formal, third-party proctored certification exam that indicates that those who have achieved it possess the in-depth knowledge to design, install, configure, maintain, and troubleshoot most ZDX implementations.

Exam Format

Certification name: Zscaler Digital Experience Administrator (ZDXA)
Delivered through: Online Proctored
Exam series: Zscaler Digital Experience
Seat time: 90 minutes
Number of items: 60
Format: Multiple Choice
Languages: English and Japanese

Audience & Qualifications

The ZDXA exam is for Zscaler customers as well as all who sell and support the Zscaler platform. By taking the exam, you are demonstrating your deep understanding and knowledge needed to sufficiently drive operational success. Candidates should have:

Experience

- One year working in both IT networks or cybersecurity frameworks
- Minimum of 3 - 6 mo experience with the Zscaler platform

Skills Required

- Platform Navigation: Ability to navigate the ZDX Admin Portal and interpret high-level dashboards (e.g., ZDX Score, User Health).
- Configuration: Proficiency in configuring application and web probes (Cloud Path, HTTP, and MTR) to monitor performance.
- Troubleshooting (MTTR): Ability to analyze metrics across the "Device, Network, and Application" layers to pinpoint root causes and reduce Mean Time to Resolution.
- Infrastructure Context: A foundational understanding of the broader Zscaler Zero Trust Exchange, specifically how ZDX integrates with

Recommended Training

Zscaler recommends that you have first attended the ZDX Operationalization (EDU-310) course and labs, or have solid hands-on experience with ZIA, ZPA and ZDX.

Exam Blueprint:

Zscaler Digital Experience Administrator

60 questions | 90 minutes | Professional Level

Prerequisites | None. *ZDX Operationalization (EDU 310) Course & Lab are strongly recommended*

The Zscaler Digital Experience Administrator exam is the final step to earn the Zscaler Digital Experience Administrator (ZDXA) certification. This online proctored exam is recommended for candidates who have a minimum of one year of experience in the Zscaler platform, including hands-on experience with ZIA, ZPA and ZDX. The course and hands-on lab, Zscaler Digital Experience Operationalization (EDU 310), help candidates prepare for the exam and are recommended prerequisites.

The following topics are general guidelines for the content likely to be included on the exam; however, other related topics may also appear on any specific delivery of the exam. In order to better reflect the contents of the exam and for clarity purposes, the guidelines below may change at any time without notice.

Monitoring and Triaging Digital Experience 30%

- Explain the 0–100 scoring logic and how it is derived from the "lowest performing metric" among Web Probes and Cloud Paths.
- Interpret the Performance Dashboard and scores to determine incident scope (single user, site, region, or widespread) and guide triage/escalation.
- Understand how ZCC interacts with ZDX cloud.
- Use Application view to compare the performance of different applications.
- Analyze device health statistics.
- Understand the basics of copilot and its usage.
- Understand the value and benefits of common third- party integrations (i.e Call Quality Monitoring, ServiceNow).
- Analyze Last Mile or Intermediate ISP performance and identify Network Anomalies.
- Understand the significance of the ZDX API for gaining programmatic access to its features.

Operational Configuration and Optimization 30%

- Given a set of requirements, configure both Predefined Applications and Custom Applications and Managed Monitoring (Hosted Monitoring) and Real User Monitoring (RUM).
- Define and differentiate between Web Probes and Cloud Path Probes and Hosted Probes.
- Given a scenario, execute approved remediation/optimization scripts at scale and validate outcomes, including prerequisites, permissions, and safety considerations.
- Given a set of requirements monitoring Real time application performance from a user perspective using browser based performance metrics.

Troubleshooting and Root Cause Analysis 30%

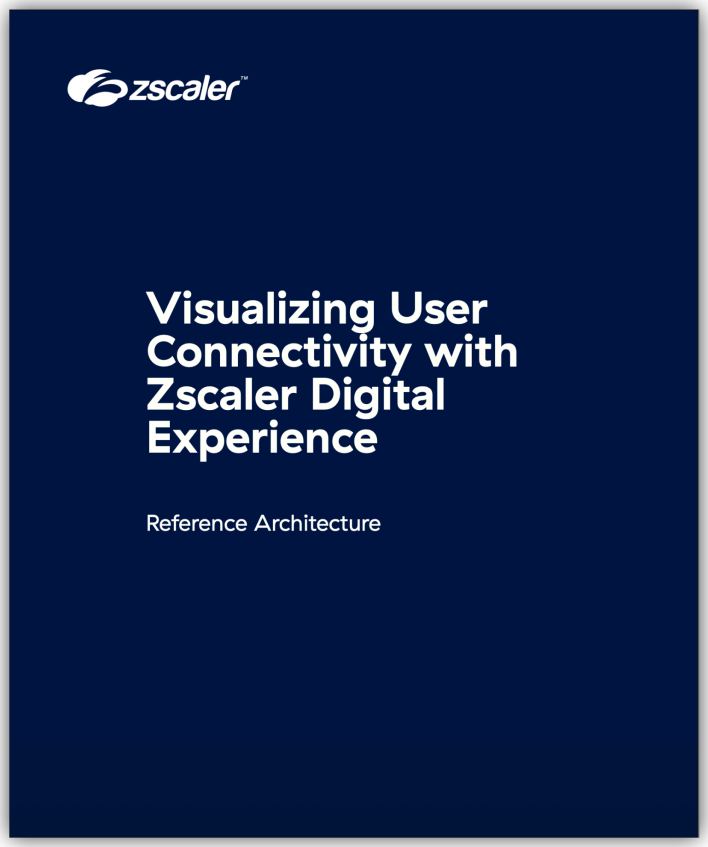
- Analyze device, network, and application telemetry to find performance issues across those tiers.
- Analyze diagnostics sessions output to determine the root cause and implement remediation.
- Analyze network path information to find bottlenecks through the entire path.
- Use Zscaler AI to analyze data for accelerated time to resolution.
- Given predefined and customized reports, analyze organization performance.
- Share current state of UI Page and generate shareable URL.
- Leverage ZDX Dashboards for organization wide incidents across various categories.

Administration and Operational Governance 10%

- Assign role based access control (e.g., giving the Helpdesk "Service Desk" access vs. "Super Admin").
- Configure Alert Rules and define trigger notification types.
- Configure admin authentication settings for ZDX.

Using the Reference Architecture to Frame Your Study

Students are encouraged to begin this study guide by reviewing "Visualizing User Connectivity with Zscaler Digital Experience - Reference Architecture" because it provides the architectural context behind the monitoring workflows covered in the ZDXA exam.



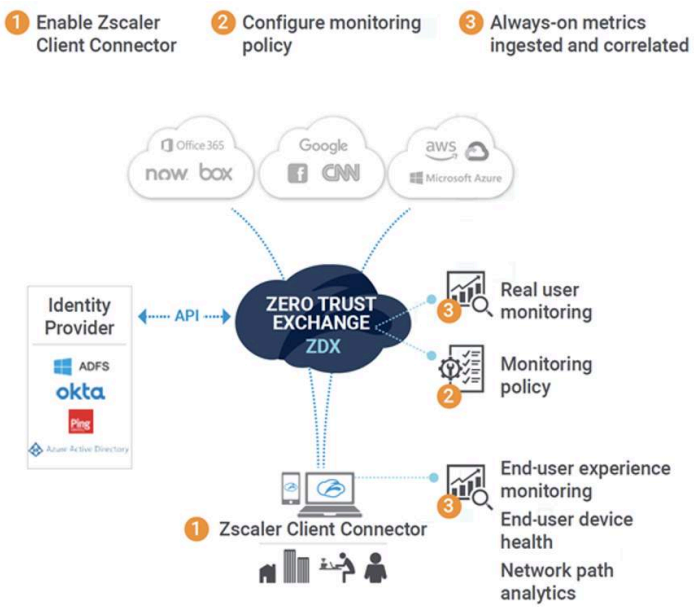
The reference architecture connects ZDX concepts such as Zscaler Client Connector telemetry, Web Probes, Cloud Path Probes, application monitoring, user experience scoring, device health, and path visibility into one coherent model. Reading it first helps candidates understand why ZDX is designed around the user's observed experience rather than a traditional data-center-only monitoring view.

Students may also benefit from revisiting the reference architecture after completing the study guide. At that point, the document becomes a useful consolidation tool: candidates can map the study guide's scenario-based troubleshooting logic back to the architecture, dashboards, probe strategy, Cloud Path interpretation, alerting, Hosted Monitoring, and operational recommendations. Used either as a starting point or a final review, the reference architecture helps turn individual ZDX features into an end-to-end mental model for monitoring, triage, and escalation.

Monitoring and Triaging Digital Experience (30%)

Monitoring and triaging digital experience is where ZDX becomes operationally useful. The administrator is not simply looking for a red number on a dashboard; the administrator is deciding whether a degraded experience is isolated to one user, concentrated in one site or region, limited to one application, or widespread across the organization. That scoping decision determines the next action, the escalation path, and whether the response should be containment, investigation, communication, or remediation.

ZDX collects experience evidence from the user perspective by using Zscaler Client Connector, synthetic application probes, Cloud Path visibility, and device telemetry. This matters because the modern application path is no longer a neat data-center path controlled end to end by IT. A single complaint might be caused by a user's endpoint, Wi-Fi, last-mile ISP, DNS path, Service Edge path, private application path, SaaS availability, or application responsiveness.



A strong ZDXA workflow begins with the broadest reasonable view and then narrows. Start with the Performance Dashboard or Applications Overview to understand the affected application and population, then move into user, application, web probe, Cloud Path, and device evidence. The goal is not to "click around until the ticket gets tired." The goal is to prove scope, correlate the low score to the metric that changed at the same time, and route the issue to the correct owner.

🎓 Exam Tip

Scope first, fix second

When a scenario says "one user," "one office," "one country," or "many users," treat that as a triage boundary. The best next step is usually the one that confirms scope and identifies the impacted users or locations before changing configuration or escalating.

Understanding the ZDX Scoring Model

The ZDX scoring model gives administrators a normalized way to compare digital experience across applications, users, locations, departments, and the organization. A score is useful because it compresses several experience signals into a form that is easy to prioritize, but it must never be treated as the whole diagnosis. The score tells you where the experience is poor; the supporting metrics tell you why.

For this exam domain, the important skill is understanding how the score is derived and how to reason from it in a scenario. Candidates should know the 0-100 model, how the lowest value affects scoring, why Page Fetch Time and availability are central to web application experience, and how Cloud Path evidence helps explain network-related degradation.

💬 Scoring Fundamentals

How the score is derived

Application:

- ZDX Score is primarily based on Page Fetch Time.
- Page Fetch Time is evaluated against regional baselines and regional peer performance (weighted average).
- Baselines are computed daily per application using the previous 7 days of data (rolling window).
- Availability failures (such as probe failures during network outages) may also reduce the score.

Department / Location / City: For each interval, take the lowest user value in that group; average interval values over the selected period.

Organization: For each interval, take the lowest application value across the org; average interval values over the selected period.

User: The user's score is the lowest application value among apps they accessed in the selected period.

The score is always interpreted in context. Time range, filters, selected applications, selected users, geolocation, operating system, last-mile ISP, and Zscaler location can all change what the score represents. The same score viewed globally, by country, by office, or for one user may point to completely different operational actions.ZDX Score Overview (0–100 Model)



The ZDX Score uses a 0-100 scale, where higher values represent better digital experience and lower values represent poorer experience. The score categories are Good from 66 to 100, Okay from 34 to 65, and Poor from 0 to 33. Scores are rounded to the nearest whole number, so

candidates should focus on score bands, trend direction, and correlated metrics rather than over-reading a one-point difference.

💬 Sidebar
Baseline-to-ZDXA bridge

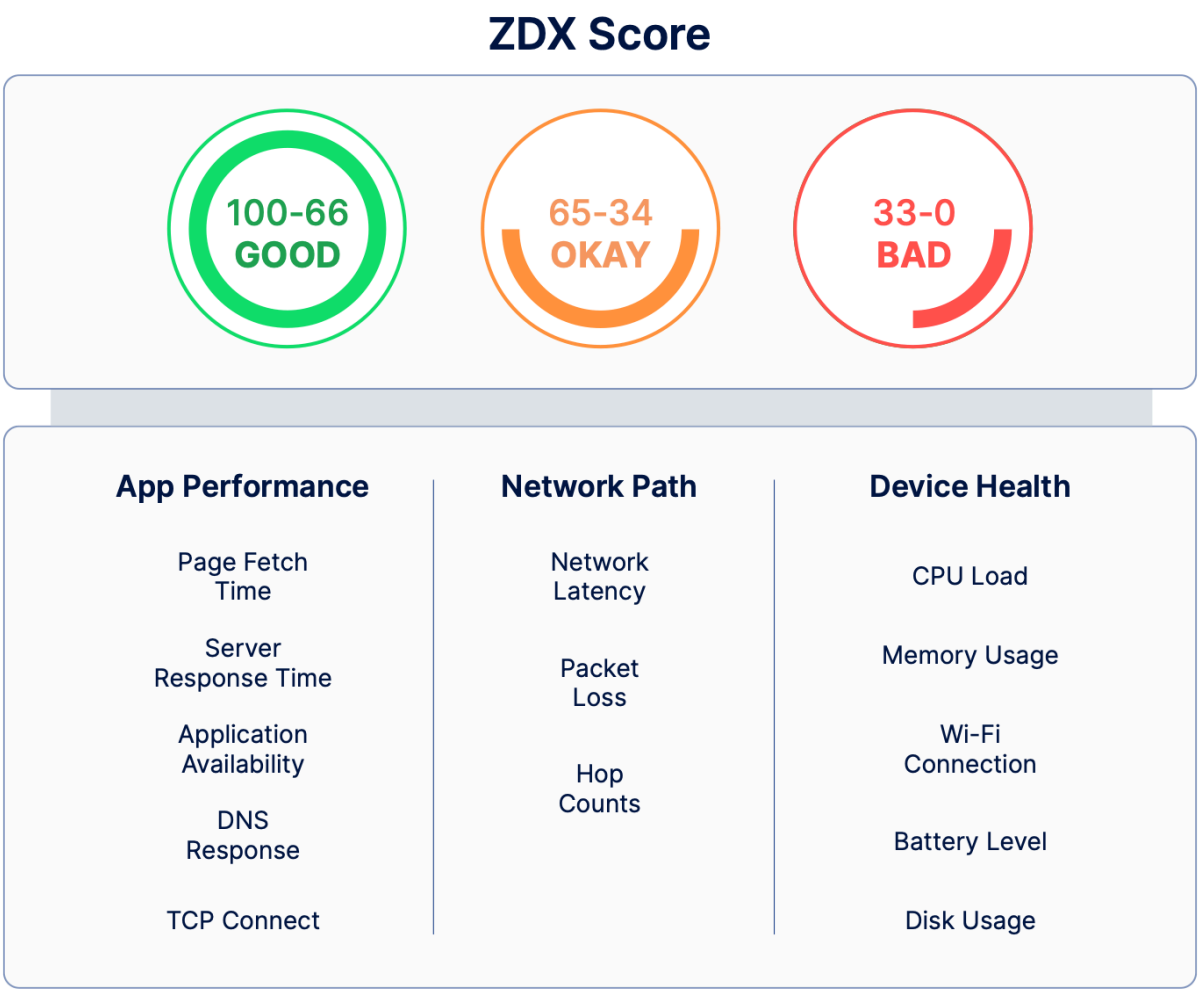
A baseline administrator can recognize that a score is low. A ZDXA-ready administrator explains what population is affected, which metric is responsible, whether the issue is current or historical, and which team should own the next action.

ZDX sends probes from Zscaler Client Connector to monitored applications at recurring intervals. For each measurement period, ZDX assigns a numerical value from 0 to 100, and the lowest value within an hour becomes the value for that hour. This is a key concept for scenario reasoning: a brief but meaningful degradation can affect the hour's displayed score even if the remainder of the hour looked healthy.

For an application score, ZDX evaluates the users who accessed that application during the selected time period and identifies the lowest value each user experienced for that application. Those lowest values are averaged to calculate the application's ZDX Score. For web applications, the application score is based primarily on Page Fetch Time compared with regional baseline behavior, and application availability can also affect the score when probes fail or the application cannot be measured successfully.

Department, location, and city scores are calculated differently from application scores. ZDX identifies the lowest value for users accessing applications from those places or groups during time intervals in the selected range, then averages the interval scores for the displayed time period. This makes those scores useful for identifying where a population is experiencing the poorest digital experience, not merely where the most users are located.

The organization score is derived by identifying the lowest application value during each time interval and then averaging those interval values over the selected time range. That means the organization score can be pulled down by the application that is hurting the organization most during a given interval. When the organization score drops, the administrator should identify which application or population caused that drop before escalating.



For a user score, ZDX compares the values across the applications the user accessed during the selected time period. The application with the lowest value represents that user's score because it reflects the user's poorest digital experience in that context. If multiple applications are selected in a user view, the displayed user experience may therefore be driven by the worst selected application rather than a comfortable average across all applications.

Baseline behavior is part of the interpretation. For Page Fetch Time, ZDX uses regional baseline information, and baseline metrics are calculated daily for each application over a rolling seven-day timeline. This is why experienced administrators look for meaningful changes from normal behavior instead of treating every absolute score the same across every region, application, and access pattern.

🎓 Exam Note

The score is a triage signal

A low ZDX Score is not a root cause by itself. It is a signal that tells you to inspect the metric stack: Page Fetch Time, availability, DNS Resolve Time, Server Response Time, Cloud Path behavior, device health, and device events.

Lowest Performing Metric Logic

The “lowest performing metric” language in the blueprint should be understood as an operational reasoning model: ZDX is designed to keep poor experience visible instead of allowing healthy metrics to hide it. In scoring terms, the lowest values within time windows and selected populations matter. In troubleshooting terms, the administrator should find the weakest correlated experience signal and use that signal to drive the next step.

This logic prevents false comfort. A user can have acceptable CPU and memory, but if Page Fetch Time is degraded or availability fails, the user still has a poor application experience. Likewise, an application can appear healthy globally while one country or office has a poor path, poor DNS behavior, or a local connectivity issue that affects only that population.

The lowest value within an hour matters for score calculation, and the lowest application or user value can influence the displayed score depending on the view. For applications, the lowest value each user experienced for that application is averaged. For the organization, the application with the lowest value in a time interval represents the organization’s score for that interval before interval values are averaged.

In practice, administrators should correlate the low score to the metric that changed at the same timestamp. If the score drops as Page Fetch Time rises, inspect Web Probe Performance. If the score drops while Cloud Path latency rises on a specific leg, inspect Hop View and Command Line View. If the score drops while CPU, memory, disk, Wi-Fi, or crash signals change, inspect user device details and device health.

The Performance Dashboard supports this reasoning by showing the Most Impacted Applications and automatically focusing the page on the selected application. The ZDX Score graph and Page Fetch Time graph can compare multiple applications over the same time window, which helps distinguish a single application issue from a shared network, provider, or regional condition.

The Regions by ZDX Score map helps translate score logic into incident scope. It displays the geographic distribution of users for the selected application and can show whether a degradation is global, regional, city-specific, or unknown-location related. When a scenario asks about a specific country or region, the administrator should use the map and filters to identify impacted locations and users before moving to deeper troubleshooting.

A scenario such as “Salesforce is slow in New Zealand” should be handled by filtering the Performance Dashboard or Applications Overview to Salesforce and the relevant geography, confirming whether the degradation is concentrated there, and then identifying the impacted users. Only after the impacted population is clear should the administrator inspect application details, Cloud Path, web probe metrics, or user device evidence.

Real User Monitoring can change what is displayed for an application. If an application has RUM enabled, the Performance Dashboard can show Real User Score and real-user trends in addition to, or instead of, synthetic monitoring trends. For this scoring section, candidates should recognize whether they are reading synthetic ZDX Score, Real User Score, or both, because each reflects a different type of evidence.

> ⚠️ **Warning**
> *Do not average away user pain*
>
> Averages can hide the experience that triggered the ticket. In ZDXA scenarios, follow the score drop to the lowest or most degraded correlated signal, then validate whether that signal explains the impacted population.

Role of Web Probes in Scoring

Web Probes measure web application availability and responsiveness by sending HTTP or HTTPS GET requests to the configured application URL. The application returns an HTTP status code, and ZDX evaluates whether that response matches the configured success criteria. This is why Web Probe configuration must reflect how the application actually behaves, especially when unauthenticated users are redirected to an identity provider.

The core Web Probe metrics are Page Fetch Time, DNS Resolve Time, Server Response Time, and Availability. Page Fetch Time measures the network fetch time for the web page from the configured probe URL. DNS Resolve Time measures how long it takes to resolve the hostname. Server Response Time represents Time to First Byte. Availability is based on whether a successful response is received; if a probe times out, availability defaults to 0.

Page Fetch Time is central to web application scoring because it approximates whether the application feels fast or slow from the user’s perspective. It requests only the top-level page document and does not request all embedded links in the page. That distinction matters: a Web Probe is not the same as a full browser waterfall, but it is a consistent synthetic measurement that can be compared across users, regions, and time.

The Web Probe Performance view allows administrators to compare the ZDX Score trend with Page Fetch Time trends. This comparison is especially useful when a low score coincides with a Page Fetch Time spike. The administrator can then inspect whether DNS, Server Response Time, or availability moved at the same time, instead of assuming that all slowness is caused by the network.

Success-code design is a common source of bad telemetry. ZDX initially monitors common informational, success, and redirection response codes, and custom web probes can be adjusted so only the response codes that represent success for that application remain valid. If the application normally redirects unauthenticated users to SSO, the redirect from the original application might be the correct availability signal. If the probe follows the redirect too far, the measurement may start validating the identity provider instead of the application front door.

Web probe requests are always TLS/SSL terminated so that the probe can be intercepted and instrumented. Configured TLS/SSL bypass policies are overridden for ZDX web probes. During a Deep Tracing session, session probe requests can also appear in ZIA web logs, which gives administrators an additional evidence source when troubleshooting internet application behavior.

Web Probe interpretation should follow the metric pattern. If Page Fetch Time and Server Response Time rise together while Cloud Path latency remains stable, the application or hosting tier becomes more likely. If DNS Resolve Time spikes during the score drop, the resolver path or DNS configuration deserves attention. If availability changes, investigate response-code behavior, timeout behavior, and reachability before assuming generic slowness.

Custom applications require careful probe design. Most applications require at least one Web Probe, while network applications that use a single Cloud Path probe use End-to-End Latency information in place of Page Fetch Time to calculate the ZDX Score. For custom web applications, define the probe URL and success criteria so the metric represents the business service users actually care about.

> 🎓 **Exam Tip**
> *Web probe elimination logic*
>
> When the scenario gives you Page Fetch Time, DNS Resolve Time, Server Response Time, and Availability, do not treat them as four synonyms for “slow.” Each metric points to a different part of the transaction.

Role of Cloud Path Probes in Scoring

Cloud Path Probes provide network-path evidence between the user’s device and the monitored destination. They map the path by sending packets with reduced time-to-live values and observing how each hop responds. This gives administrators hop-by-hop visibility into the path that the user experience depends on.

Cloud Path Probes can use ICMP, UDP, or TCP, and the protocol can vary by segment when Adaptive behavior is used. The resulting data helps identify whether the path is healthy, whether latency is concentrated in a specific leg, and whether a network hop or segment appears to contribute to the user’s degraded experience.

The main Cloud Path metrics are Hop Count, Packet Loss, and Latency. Latency includes average, minimum, maximum, and standard deviation measurements. In Application Details, Network Probe Performance can also show End-to-End latency, packet loss, total hops, and packet count (max of 11). These metrics help identify whether a poor experience is related to network path behavior rather than application response behavior.

Cloud Path evidence is presented through multiple views. The latency or packet-loss graph is best for time correlation. Hop View is best for understanding where the degraded leg sits in the path. Command Line View is best for detailed inspection of hop addresses, path details, and measured values. A ZDXA candidate should know which view to use for the question being answered.

The path differs by application delivery model. A direct path generally shows the user device, local egress, and destination. A ZIA path adds the ZIA Service Edge segment. A ZPA path adds the ZPA Service Edge and App Connector side of the path. These distinctions matter because remediation ownership changes depending on whether the degradation is local, ISP-related, Service Edge-related, private access-related, or application-side.

Adaptive mode is useful because internet devices do not always respond consistently to a single probing protocol. Adaptive mode tries TCP, UDP, and ICMP and selects the best available protocol for each Cloud Path leg based on factors such as latency and loss. For protocol-specific application routing or custom forwarding, the administrator may need a more specific protocol and port so that the probe follows the path that matters.

Intermediate hop data requires discipline. A high value on a middle hop does not always mean that hop is harming the user experience, because intermediate devices can rate-limit or deprioritize probe responses. Strong analysis checks whether the destination, the path leg, and the application experience show the same degradation pattern before concluding that a middle hop is the cause.

For UCaaS Call Quality applications such as Microsoft Teams, Zoom, and Webex, Cloud Path evidence is especially important because Web Probes are not allowed for those Call Quality applications. ZDX can use Autosense Cloud Path behavior to dynamically discover a call destination IP address and provide automated Cloud Path probing. Call Quality scoring can be based on MOS when latency, jitter, and packet loss are available, or on metric thresholds when only one or two of those metrics are available.

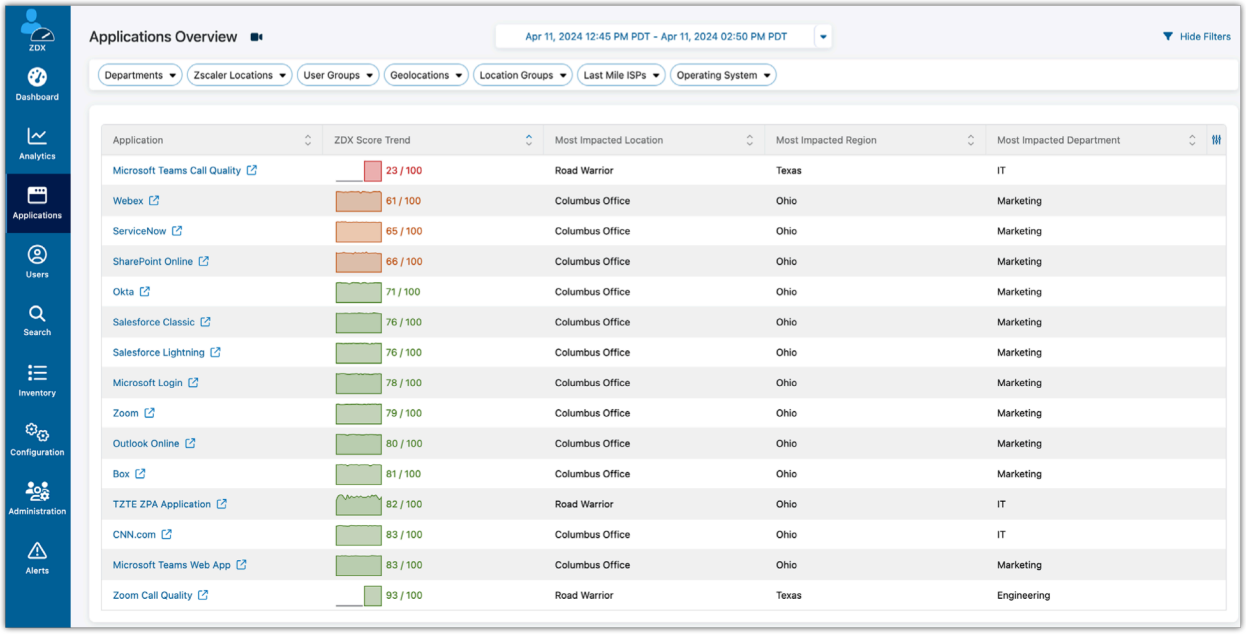
When a score drop aligns with increased Cloud Path latency between the user’s egress and a Service Edge, the network path deserves investigation. When Cloud Path remains stable while Server Response Time rises, application responsiveness becomes more likely. ZDXA scenarios reward this layered interpretation; they do not reward blaming “the cloud” like it is a magic fog machine.

> ⚠️ **Warning**
> *Intermediate hop trap*
>
> Do not escalate based only on a noisy middle hop. Confirm whether the destination or affected path leg shows degradation and whether the timing aligns with the user's score drop.

Interpreting Application vs User vs Device Scores

Application scores answer the question, "How is this application performing for the selected population?" Applications Overview and Application Details provide the population-level view, including score trends, web probe performance, network probe performance, impacted geolocations, impacted departments, impacted regions, and impacted locations. If RUM is enabled, Application Details can also include Real-User Performance and Errors tabs.

The Applications Overview is the right starting point when multiple users report the same business service. It allows comparison across applications and can show whether one monitored application is degraded while others remain stable. In Application Details, the administrator can compare ZDX Score and Page Fetch Time, inspect probe status, and review the top impacted regions, departments, and locations.



Application	ZDX Score Trend	Most Impacted Location	Most Impacted Region	Most Impacted Department
Microsoft Teams Call Quality	23 / 100	Road Warrior	Texas	IT
Webex	61 / 100	Columbus Office	Ohio	Marketing
ServiceNow	65 / 100	Columbus Office	Ohio	Marketing
SharePoint Online	66 / 100	Columbus Office	Ohio	Marketing
Okta	71 / 100	Columbus Office	Ohio	Marketing
Salesforce Classic	76 / 100	Columbus Office	Ohio	Marketing
Salesforce Lightning	76 / 100	Columbus Office	Ohio	Marketing
Microsoft Login	78 / 100	Columbus Office	Ohio	Marketing
Zoom	79 / 100	Columbus Office	Ohio	Marketing
Outlook Online	80 / 100	Columbus Office	Ohio	Marketing
Box	81 / 100	Columbus Office	Ohio	Marketing
TZTE ZPA Application	82 / 100	Road Warrior	Ohio	IT
CNN.com	83 / 100	Columbus Office	Ohio	IT
Microsoft Teams Web App	83 / 100	Columbus Office	Ohio	Marketing
Zoom Call Quality	93 / 100	Road Warrior	Texas	Engineering

User scores answer the question, "What is this user's poorest digital experience in the selected context?" The Users Overview requires application selection before user score distributions and user lists become meaningful. It organizes users into Good, Okay, and Poor score categories and provides a path into User Details for device, application, web probe, Cloud Path, and event analysis.

In User Details, the selected application matters. If a user has multiple applications in scope, the application with the lowest score can drive what the administrator sees first. This is helpful for troubleshooting because it surfaces the user's worst experience, but it also means the administrator must confirm which application is actually selected before drawing conclusions.

Smooth ZDX Score helps reduce short-term variation and makes trends easier to read. It is useful for pattern recognition, especially when raw scores are noisy. For a precise troubleshooting decision, however, the administrator should still inspect the raw ZDX Score and the timestamped supporting metrics around the reported issue.

Device Health Score is related to user experience but is not the same thing as an application ZDX Score. Device Health uses endpoint contributing factors such as CPU usage, memory usage, disk usage, average disk queue length, battery, Wi-Fi signal quality, system crashes, and software crashes. It has the same Good, Okay, and Poor style of score bands, but it measures device health rather than application experience.

Device Events Reports provide aggregated insight into system crashes and software crashes that can affect user experience. System crashes can include BSOD events and unexpected shutdowns or reboots, while software crash reporting can include crashes and hangs. These reports help administrators determine whether a poor user experience is tied to a device condition that occurred during the same time window.

The correct escalation path depends on the score lens and supporting evidence. A broad application score drop across many regions may belong to the application owner or SaaS provider. A regional score drop with path degradation may belong to network or service-provider investigation. A single-user issue with CPU, Wi-Fi, crash, or device event correlation belongs first in user/device troubleshooting.

Do not conflate device health with application health. A single user with a poor device can have a poor experience while the application score remains healthy for everyone else. Conversely, a widespread application or path issue can produce poor user scores even when individual devices are healthy. The administrator's job is to separate those cases before assigning ownership.

In a single-user scenario, begin with the user's selected application, device, Web Probe metrics, Cloud Path, and device events. In a regional scenario, begin with Performance Dashboard or Applications Overview filters, then use impacted users and Cloud Path evidence to identify whether the region shares a last-mile, ISP, Service Edge, or application pattern. That difference is the gap between triage and ticket pinball.

🔄 Cross-policy resources

ZDX, ZIA, ZPA, Client Connector

ZDX interpretation depends on the application delivery path. SaaS through ZIA, private applications through ZPA, direct access, and UCaaS Call Quality applications can show different probe behavior and different Cloud Path legs. Client Connector provides the user-device perspective that ties those paths back to the actual experience.

📝 Quick Review

- Start with scope: single user, site, region, application, or organization-wide.
- Use the ZDX Score as a triage signal, not as the root cause.
- Good is 66-100, Okay is 34-65, and Poor is 0-33.
- The lowest value within an hour affects the displayed hourly score.
- Application scores use the lowest value each user experienced for that application and average those values.
- User scores reflect the user's poorest selected application experience.
- Web Probes explain application fetch behavior, DNS resolution, server response, and availability.
- Cloud Path Probes explain network path behavior, including path legs, hop count, latency, packet loss, and destination path visibility.
- Device Health Score is separate from application ZDX Score; correlate it before assigning blame.
- Common mistake: treating a low score as proof of an application outage.
- Common mistake: blaming an intermediate hop without validating the destination, path leg, and timestamp.
- Common mistake: ignoring filters and comparing scores from different populations or time ranges.
- Common mistake: reading Smooth ZDX Score as the only evidence during a precise incident investigation.

🏅 Exam Blueprint Alignment

Scenario Review

- Explain the 0-100 scoring logic and how ZDX Scores are derived from low-performing experience values.
- Interpret the Performance Dashboard and scores to determine whether an issue affects a single user, site, region, or broader population.
- Use Applications Overview and Application Details to compare application performance across regions, departments, and locations.
- Analyze device health statistics to determine whether endpoint health or local connectivity is contributing to poor experience.
- Analyze last-mile and intermediate ISP performance using Cloud Path evidence and supporting dashboard filters.
- Understand how Zscaler Client Connector contributes synthetic probes, device metrics, and user-side visibility to ZDX.
- Use score trends, impacted users, impacted locations, and supporting metrics to guide triage and escalation.

Using the Performance Dashboard for Triage

The Performance Dashboard is the best starting point when the complaint is still broad and the engineer needs to establish scope. It provides an organization-level view of application experience, regional distribution, score trends, and Page Fetch Time behavior for monitored applications. Its job is not to complete root cause analysis; its job is to tell the engineer where to go next.

For the ZDXA exam, this subsection is scenario-heavy. Candidates should expect to interpret a described condition, apply the correct dashboard filter or comparison, and decide whether to pivot into Users Overview, Applications Overview, Incidents, User Details, Diagnostics, or an escalation path. The skill being tested is not “where is the widget?” It is “what does this pattern mean?”

A clean triage workflow starts by selecting the reported application and the correct time range, then checking whether the degradation is concentrated by geography, Zscaler location, user group, department, last mile ISP, or operating system. From there, the engineer decides whether the incident is isolated, clustered, regional, application-specific, or broader than a single application.

Performance Dashboard Overview

The Performance Dashboard provides an overview of ZDX experience for the organization during the selected time range. In practice, it helps answer four first-response questions: which application is most affected, where the poor experience is concentrated, when the decline occurred, and whether Page Fetch Time aligns with the score change.

Most Impacted Applications is the first application-level signal. It surfaces applications with lower experience scores in the selected period and keeps the dashboard focused on the selected application. In a scenario, selecting the reported application is usually the first step because it prevents unrelated applications from diluting the view.

The ZDX Score graph shows how the selected application’s score trends over time. Good, Okay, and Poor bands are useful for orientation, but the engineer should primarily look for meaningful change. A sudden score drop during the complaint window is more actionable than a score that has always been lower for a known high-latency geography.

The Page Fetch Time graph tracks how long the selected application takes to transfer the fetched page during the selected period. Page Fetch Time is a key web-experience signal because a poor fetch time can degrade the user’s experience even when other indicators look acceptable. The P95 line helps distinguish ordinary user variation from outlier behavior by showing the level below which most fetch times fall.

The Regions by ZDX Score map adds geographic context. It shows where users are experiencing the selected application and helps reveal whether the issue is concentrated in a city, country, region, or spread across unrelated locations. This is often the fastest way to distinguish a regional issue from an application-wide one.

Time range selection must match the operational question. A current view is useful for “is it happening now?” Longer and custom ranges are better when the ticket references a past event, a business-hour window, or a change that occurred earlier in the day. If the time range is wrong, every conclusion downstream is suspect.

A cleaner way to read the dashboard is to compare the score trend, Page Fetch Time, and map distribution together. If the score drops and Page Fetch Time rises in one region, the engineer has a strong regional web-performance lead. If the score drops across multiple applications at the same time, the issue may involve a shared path, endpoint population, location, ISP, or platform dependency rather than one application.

🎓 Exam Tip

Read the dashboard as a scope detector

Start with the reported application and time window. Then use the score trend, Page Fetch Time, and map distribution to decide whether to investigate users, locations, application details, incidents, or network path evidence.

Identifying Incident Scope (User, Site, Region, Global)

Incident scope is the most important triage decision because it determines who owns the next action. A single-user issue usually belongs in user and device analysis. A site issue may involve Wi-Fi, local egress, Zscaler location context, or last mile ISP behavior. A regional issue may involve geolocation, ISP patterns, or application behavior affecting a geography. A global issue requires broader validation before any team is blamed.

A single-user scope is indicated when one user or one device is poor while other users in the same application, location, or region remain healthy. The correct pivot is User Details, where device information, selected application score, web metrics, network performance, device events, and Diagnostics can be evaluated together.

A site-level scope is suggested when multiple users associated with the same Zscaler location, location group, office, SSID, or local egress pattern show similar degradation in the same period. In that case, the engineer should resist escalating directly to the application owner until site-specific evidence is reviewed.

A regional scope appears when poor experience is concentrated in a country, city cluster, or geography while other regions remain healthier for the same application. The Performance Dashboard map and geolocation filters help establish this pattern. The next step is usually to identify impacted users and locations, then review whether the pattern follows ISP, network path, application, or Service Edge evidence.

An application-wide scope is suggested when one application is degraded across unrelated locations, user groups, and regions while other applications remain stable. Application Details

becomes the better next view because the engineer needs to inspect application-specific score trends, web probe behavior, availability, server response behavior, and affected populations.

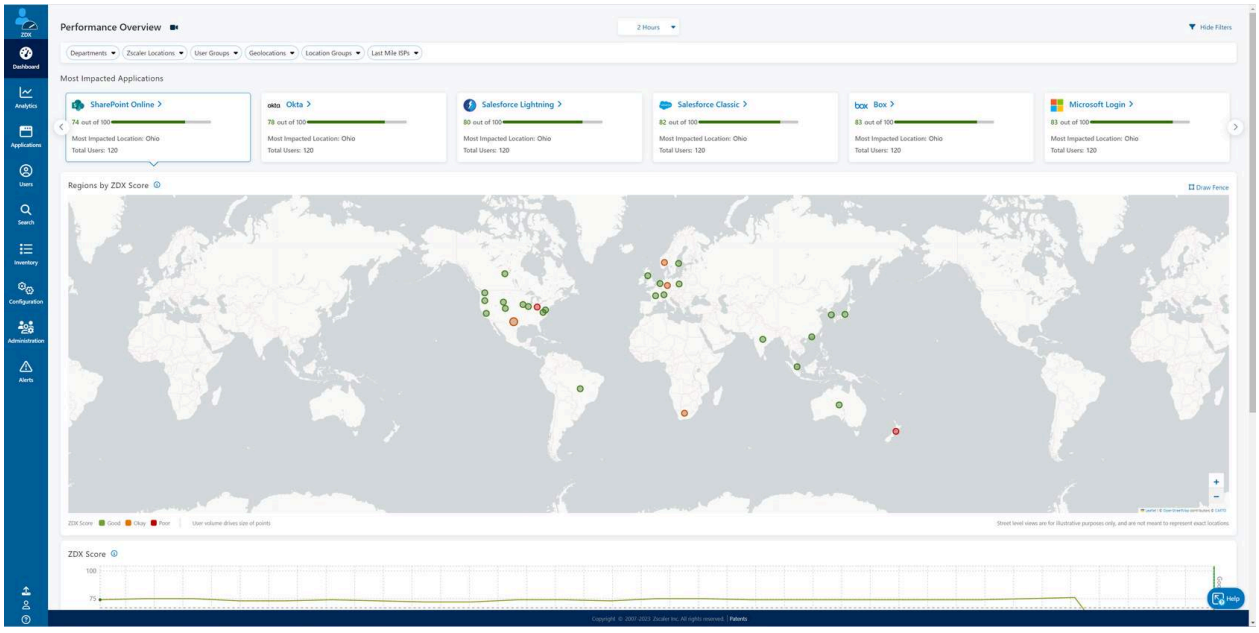
A widespread scope is suggested when several unrelated applications degrade during the same period. The engineer should compare additional applications on the score and Page Fetch Time graphs before assuming each application has a separate problem. Multiple applications declining together often points to shared infrastructure, access path, endpoint cohort, ISP behavior, or platform dependency.

The Incidents Dashboard can validate whether ZDX has detected a multi-user incident in categories such as device, Wi-Fi, last mile ISP, intermediate ISP, ZIA Public Service Edge, ZPA, or application. Incidents should be used as acceleration evidence, not as a substitute for reasoning. The best workflow is to use the Performance Dashboard to establish the pattern, then use Incidents to confirm whether ZDX has already correlated the affected area.

> ⚠ **Warning**
> *Most impacted does not mean root cause*
>
> A most impacted application, region, or location is a triage lead. It is not proof that the application, region, or location is the failing component. Validate the layer before routing the escalation.

Filtering by Application, Region, and User Groups

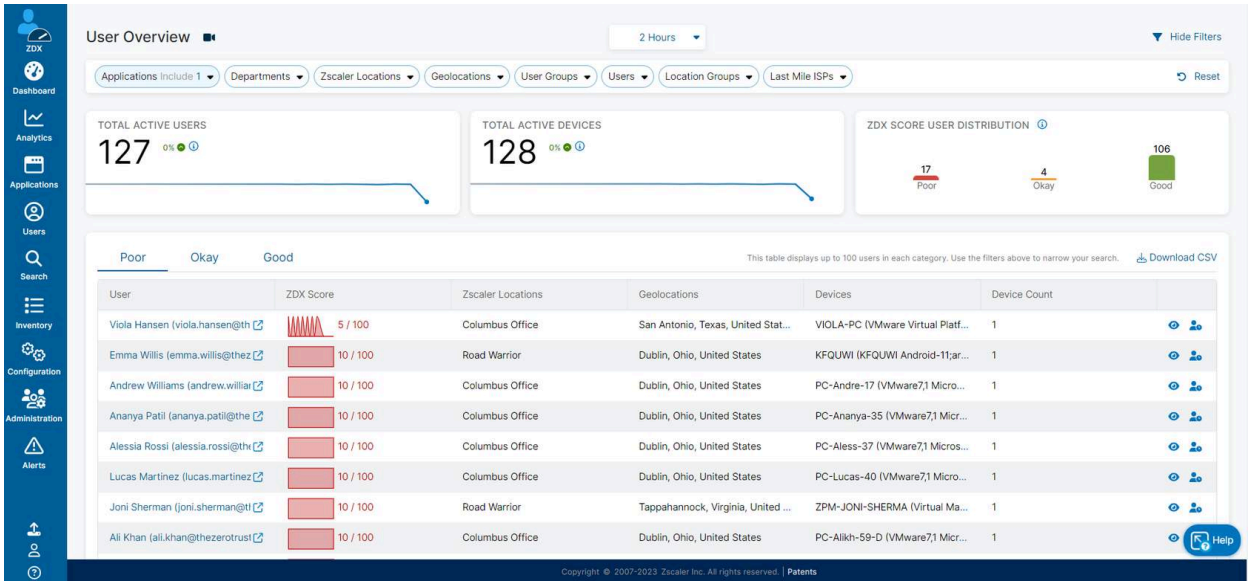
Filtering is the method for proving or disproving a scope hypothesis. The cleanest pattern is to filter narrowly enough to isolate the reported symptom, but not so narrowly that the dashboard hides related users. Overfiltering can make a real incident look like a one-off ticket.



Application filtering usually comes first because most user complaints are expressed in business terms: “Outlook is slow,” “Salesforce is timing out,” or “ServiceNow is unusable.” Selecting the application aligns the dashboard with the user’s symptom and prevents unrelated application telemetry from confusing the investigation.

Geolocation and region filters help determine whether the problem follows a geography. If a single country or city cluster shows degraded experience while other regions remain healthy, the engineer should continue narrowing by location, ISP, and impacted users. If many regions show similar degradation, the application or shared path deserves more attention.

User group and department filters are useful when the affected population follows identity or business function instead of geography. For example, an application may look acceptable overall but perform poorly for a department that uses a different tenant, private application path, or business workflow. In that case, group-based filtering is more precise than a map-first approach.



Zscaler locations and location groups are useful for separating office or branch behavior from roaming-user behavior. If the issue follows a defined location, local access, egress, or site-specific network conditions become stronger candidates. If the issue follows a user group across many locations, the engineer should consider application assignment, device cohort, user profile, or shared service behavior.

Last mile ISP and operating system filters are secondary but powerful. Last mile ISP filtering helps with remote-work patterns where users share a provider rather than a corporate site. Operating system filtering helps when the issue may follow a device platform, software rollout, Client Connector version, or endpoint change.

The best filter strategy is iterative. Apply one meaningful filter, observe whether the signal sharpens or disappears, then decide the next filter. Candidates should avoid stacking every

available filter at once. That approach may feel thorough, but it can remove the comparison population needed to understand the issue.

Identifying Impacted Users and Locations

After scope is established, the engineer must identify who is impacted and where they are located. This step converts dashboard interpretation into operational evidence. It supports ticket correlation, user communication, escalation routing, and containment decisions.

The Regions by ZDX Score map is the fastest way to identify affected geography for a selected application. When the map shows a concentrated cluster, the engineer should move from “this region looks degraded” to “which users and devices are affected?” That pivot usually leads to Users Overview with the relevant application and location context preserved.

Users Overview provides the user-level view for the selected application and filters. It shows users by score category and provides access to user and device details. This is the correct next step in a scenario where the Performance Dashboard confirms that an application is degraded in a specific region and asks how to identify the affected users.

Score categories help prioritize response. Poor users are the immediate operational concern, Okay users may show emerging degradation, and Good users provide a useful control population. A healthy control population is valuable because it proves the issue is not universal and helps isolate the difference between affected and unaffected users.

Device count matters because users and devices are not the same object. A user with one poor device and another healthy device suggests a device-specific issue. Multiple users with poor devices in the same location suggest a location or access issue. Multiple devices for one user on the same poor network suggest local connectivity or ISP behavior.

Application Details provides a service-centered view of impact. It helps identify impacted departments, regions, locations, geolocations, and probe metrics for the selected application. This view is especially useful when the engineer needs evidence that one application is affecting multiple populations before escalating to an application owner.

Incident details provide another impact view when ZDX has detected a correlated incident. The incident view can show the incident type, epicenter, severity, impacted users, impacted geolocations, affected applications, top impacted users, and key metrics appropriate to the incident category. This helps the engineer separate “many tickets” from “one correlated incident.”

Location accuracy should be interpreted with care. ZDX can use device location services when available and can fall back to IP-based geolocation when location services are unavailable. For triage, that is usually good enough to find patterns, but engineers should validate affected users, devices, and locations before making a high-impact routing decision.

Treat the impacted population as part of the evidence package. A useful triage note includes the application, time range, affected geography or location, number or pattern of impacted users,

score behavior, Page Fetch Time behavior, and any incident category observed. That is the difference between an escalation that lands and an escalation that boomerangs.

💬 Sidebar
Impact is a routing input

The question is not only “what is broken?” It is also “who is affected, where are they, and which team can act on that evidence?” Scope plus impact drives the correct owner.

Escalation Decision Framework

Escalation should follow evidence, not organizational reflex. The Performance Dashboard helps determine whether the next owner is endpoint support, network operations, application operations, ZIA operations, ZPA operations, or incident management. A strong escalation states what is known, what was tested, and what remains uncertain.

Escalate toward endpoint or service desk support when the issue is isolated to one user or one device and the evidence points to local device or access conditions. Examples include device health symptoms, weak Wi-Fi indicators, recent device events, or a poor experience that does not appear for other users of the same application in the same area. The next operational step is usually User Details and Diagnostics.

Escalate toward network operations when the issue clusters by site, Wi-Fi, last mile ISP, intermediate ISP, or route segment. The engineer should include the affected location, user population, time window, and relevant latency or packet-loss evidence. If the issue follows an ISP or site rather than an application, application escalation is premature.

Escalate toward application owners when one application is degraded across unrelated users, regions, and locations while other applications remain healthy. Web probe evidence strengthens this escalation, especially when Page Fetch Time, server response behavior, availability, HTTP errors, redirects, or application-specific incident metrics support an application-side pattern.

Escalate toward ZIA operations when evidence points to Internet & SaaS Service Edge behavior, Service Edge connectivity, ZIA-related incident metrics, or a shared ZIA path pattern. The Performance Dashboard alone is not enough to prove this. The escalation should be supported by incident details, application details, and network or transaction evidence that indicates the ZIA layer is relevant.

Escalate toward ZPA operations when the issue is specific to private application access, ZPA Service Edge behavior, App Connector behavior, or private application path evidence. A private application complaint can still be caused by endpoint, access network, app server, connector, or service path conditions, so the escalation should state which layer the evidence currently supports.

Escalate into incident management or automation when the affected population and severity justify coordinated response. ZDX alerting, webhook-based routing, and ServiceNow integration can help operationalize this workflow when configured. The administrator-level governance principle is least privilege: integrations and service users should have the access needed for the workflow, not broad administrative authority because it is “just easier.”

A good escalation preserves uncertainty without being vague. “Salesforce degraded for users in New Zealand during the selected window; Page Fetch Time increased; impacted users are concentrated in that geography; other regions are healthier” is actionable. “Salesforce is bad” is not. The dashboard gives the engineer the evidence to avoid the second version.

🔄 Cross-policy resources

ZDX evidence, adjacent action

ZDX often identifies evidence that belongs to adjacent operational domains: ZIA, ZPA, Client Connector, identity groups, location definitions, endpoint support, application ownership, logging, or ITSM workflows. Keep ZDX central as the evidence source, then route to the adjacent domain only when the evidence supports that handoff.

📝 Quick Review

- Start with the reported application and the complaint time window.
- Use the Performance Dashboard to determine scope before opening individual user details.
- Treat ZDX Score as a symptom, not a root cause.
- Correlate score change with Page Fetch Time, geography, impacted users, and incident evidence.
- Use filters iteratively: application, time range, geography, location, user group, department, ISP, and operating system.
- Use the map to identify affected regions, then pivot to Users Overview to identify impacted users and devices.
- Use Application Details when one service appears degraded across multiple populations.
- Use Incidents when ZDX has correlated a multi-user condition into an incident category.
- Escalate to endpoint support for isolated user or device evidence.
- Escalate to network operations for site, Wi-Fi, ISP, or path-based evidence.
- Escalate to application owners when one application is degraded across unrelated users and locations.
- Escalate to ZIA or ZPA operations only when the evidence points to the relevant Service Edge, App Connector, or access path.
- Avoid common mistakes: wrong time range, overfiltering, treating “most impacted” as root cause, ignoring device count, and escalating before identifying affected users.

🏅 Exam Blueprint Alignment

Scenario Review

- Section 1.1: Explain the 0–100 scoring logic and interpret Good, Okay, and Poor score bands when using dashboard views for triage.
- Section 1.2: Given a scenario, interpret the Performance Dashboard and scores to determine whether incident scope is a single user, site, region, or widespread issue, then guide triage or escalation.
- Section 1.4: Use application views and dashboard comparisons to evaluate performance across different applications.
- Section 1.5: Analyze device health context when the dashboard signal narrows to a user or device-level issue.
- Section 1.8: Recognize when last mile or intermediate ISP patterns may explain degraded regional experience.
- Section 3.1: Analyze application, device, and network telemetry together instead of relying on a single score.
- Section 3.3: Use network-path evidence when latency or packet loss becomes the likely bottleneck.
- Section 3.7: Leverage dashboards for organization-wide incidents across categories such as device, Wi-Fi, ISP, Service Edge, ZPA, and application.

Zscaler Client Connector (ZCC) and ZDX Cloud Interaction

Zscaler Client Connector (ZCC) is the endpoint-side collection point for ZDX. It performs synthetic monitoring, collects device health signals, contributes Cloud Path observations, and exposes local service state that helps administrators determine whether the endpoint is producing valid ZDX telemetry. This interaction is central to understanding how ZCC and the ZDX cloud work together.

This topic is best studied as an evidence chain rather than a static architecture component. In a scenario, the issue may be missing ZDX data, stale metrics, a sudden score drop, an unexpected geolocation, or different results for users accessing the same application. The correct reasoning depends on connecting ZCC health, entitlement, probe scope, policy propagation, cloud ingestion, and dashboard interpretation.

ZCC also participates in ZIA and ZPA access paths, which affects how ZDX evidence is interpreted. A public SaaS application reached through a ZIA Public Service Edge produces a different Cloud Path pattern from a private application reached through a ZPA Service Edge and App Connector. The exam-relevant skill is not naming those components; it is recognizing which segment owns the symptom.

🎓 Exam Note

Baseline-to-engineer bridge

At the baseline level, it is enough to know that ZDX uses Zscaler Client Connector. At the ZDXA level, candidates should connect endpoint telemetry, monitoring policy, cloud ingestion, analytics, and dashboard interpretation into one operational workflow.

ZCC Telemetry Collection

ZCC collects telemetry from the user’s device so ZDX can measure experience from the user perspective. This matters because users often work from locations outside direct enterprise control, including home networks, branch locations, public networks, and mobile environments. ZDX is valuable because it observes the path and device that the user actually depends on.

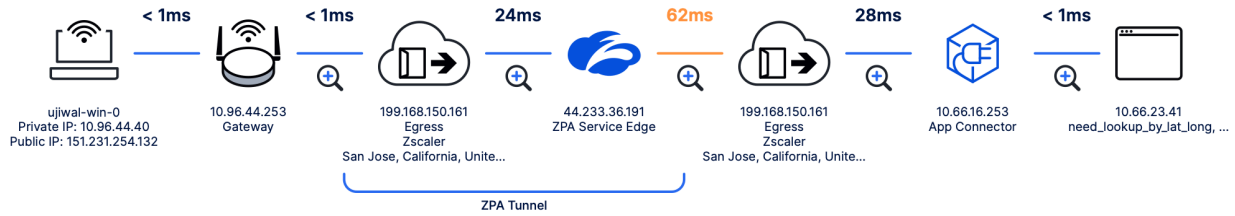
For web application monitoring, ZCC runs Web Probes against configured applications. These probes measure application availability and responsiveness using metrics such as Page Fetch Time, DNS Resolve Time, Server Response Time, and response-code-based availability. These metrics help separate slow page retrieval, DNS delay, slow server response, and application unreachability.

A Web Probe should be interpreted as a transaction measurement, not a full browser-session replay. Page Fetch Time measures the network fetch time for the URL specified in the probe. When Page Fetch Time rises, candidates should compare related metrics such as DNS Resolve

Time and Server Response Time before assigning the cause to the application, DNS, or the path.

For network-path monitoring, ZCC runs Cloud Path Probes. These probes collect hop count, latency, packet loss, and path detail between the user device and the destination. Cloud Path evidence helps answer one of the most important triage questions: where along the path does the degradation begin?

Cloud Path Probes can use ICMP, UDP, TCP, or Adaptive mode where supported. Adaptive mode can select the best protocol per leg by testing available protocols and using the most suitable one for the path segment. The candidate should read the resulting path evidence by segment, not as a single undifferentiated “network is bad” conclusion.



The same Cloud Path evidence has different operational meaning depending on the access route. A direct SaaS path, a ZIA path through a Public Service Edge, and a ZPA path through a Service Edge and App Connector each imply different escalation owners. The segment showing abnormal latency, packet loss, or probe errors should drive the next action.

ZCC also contributes device health telemetry. CPU usage, memory usage, disk behavior, network I/O, Wi-Fi signal quality, and related endpoint metrics can show whether a poor experience is device-driven. A user with high CPU utilization and normal application and Cloud Path metrics should not be escalated as an application outage.

Device events provide timeline context that raw metrics may not explain on their own. Network changes, Wi-Fi changes, restarts, sleep or wake events, and other endpoint state changes can correlate with a ZDX Score change. The strongest troubleshooting conclusion usually aligns the user-reported time, score change, metric spike, and device event.

The Digital Experience view in ZCC provides local evidence for ZDX health. Useful indicators include service status, authentication status, server address, time connected, and ZDX service version. Actions such as clearing ZDX data or restarting the ZDX service should be used as targeted troubleshooting steps after confirming that the endpoint-side ZDX service appears impaired.

Telemetry collection depends on entitlement and monitoring scope. If a user is not enabled for ZDX, is outside the probe criteria, or is not assigned to the relevant application monitoring policy, missing telemetry can be expected behavior. This is a common scenario trap: missing ZDX data is not always a broken data path.

> ⚠️ **Warning**
> *Missing telemetry is still evidence*
>
> When ZDX data is absent, stale, or inconsistent, do not immediately blame the application or ZDX cloud. First validate ZCC service state, authentication, version compatibility, ZDX entitlement, and probe assignment. The cleanest troubleshooting move is often proving that the endpoint is supposed to be collecting the data at all.

Data Flow from Endpoint to ZDX Cloud

The ZDX data flow begins with monitoring configuration. Administrators define applications, probes, criteria, exclusions, alerts, and related settings. After those changes are saved and activated, the ZDX control plane makes the applicable monitoring policy available so entitled endpoints can begin collecting the expected telemetry.

Activation and policy propagation matter in troubleshooting. A newly configured probe or application does not become useful simply because it exists in the configuration interface. It must be saved, activated, and received by the appropriate endpoint population before administrators should expect telemetry to appear.

Once ZCC receives its monitoring policy, it performs the assigned collection. Depending on configuration and feature availability, that collection may include Web Probe results, Cloud Path Probe results, device health telemetry, device events, diagnostics data, and inventory-related data. ZCC is therefore both the measurement point and the sender of endpoint-side evidence.

The Telemetry and Policy Gateway is the exchange point between ZCC and the ZDX cloud. It receives monitoring metrics from ZCC and provides monitoring policy updates back to the endpoint. From an administrator perspective, this explains why ZDX is not only a data upload path; it is a bidirectional monitoring and policy interaction.

ZDX Central Authority provides the central control-plane function for ZDX. It supports policy and configuration coordination and uses Zscaler platform context such as users, groups, departments, locations, and devices where applicable. This is why identity quality, provisioning accuracy, and group membership can directly affect what ZDX monitors and how dashboards can be filtered.

ZDX Analytics processes and aggregates received telemetry into the operational views used in Experience Center. Organization-level, application-level, location-level, and user-level views all depend on the underlying telemetry being collected and correlated correctly. If upstream collection is incomplete, downstream dashboard conclusions should be treated carefully.

The ZDX Score summarizes digital experience on a 0–100 scale. Good, Okay, and Poor ranges help administrators prioritize issues, but the score is still a summary. The cause of a poor score

must be inferred from the supporting Web Probe, Cloud Path, device health, and event evidence.

Application, user, department, location, and organization views do not all summarize experience in the same way. A user view may highlight the poorest application experience for that user in the selected time range, while application and location views aggregate across users and time intervals. Candidates should interpret scores in the context of the dashboard where they appear.

The Performance Dashboard is the normal starting point for scoping. If an application is degraded across many geographies, escalation may point toward the application or a widespread dependency. If degradation is concentrated in one country, city, office, or last-mile ISP, the next step is to drill into affected users and path evidence.

Geolocation is useful, but it is still a derived signal. ZDX can use device location information when available and can fall back to IP-based location when device location is unavailable. If a user appears in an unexpected city or in an unknown location, location source and accuracy should be considered before making a hard operational conclusion.

🎓 Exam Tip

Use the dashboard as the starting grid

Use the Performance Dashboard to establish scope, Applications views to evaluate application impact, User Details to validate a specific user's experience, Cloud Path to isolate path degradation, and device health to rule in or rule out endpoint contribution. That sequence keeps triage disciplined without turning the workflow into a click-path exercise.

Dependencies on ZCC Health and Configuration

ZDX depends on supported ZCC and ZDX module versions. If the endpoint is running an unsupported or incompatible version, certain ZDX features may not collect, display, or operate as expected. Version compatibility is therefore an operational prerequisite, not a housekeeping detail.

ZDX also depends on service entitlement. Administrators can enable ZDX broadly or selectively for groups, depending on rollout strategy and monitoring requirements. If one user population has telemetry and another does not, entitlement and group membership should be checked before assuming a network or cloud-side failure.

Application and probe design must align with the business use case. Predefined applications can simplify monitoring because associated probes are already available, while custom applications require administrators to define the monitoring structure. In either case, the key design question is whether the right users are assigned to the right probes.

Web Probes and Cloud Path Probes answer different questions. Web Probes measure application availability and web transaction responsiveness. Cloud Path Probes measure path

behavior such as latency, packet loss, and hop information. For most application monitoring, the Web Probe is foundational; Cloud Path adds the network-path explanation.

Criteria and exclusions determine which devices run probes. User groups, users, departments, locations, location groups, and devices can be used to scope monitoring. Good design avoids unnecessary probe assignment, especially for applications used only by a specific department, geography, or role.

ZDX requires required network and service communication to be allowed. If ZCC cannot reach the necessary ZDX services or related Zscaler resources, telemetry and policy exchange can fail or become incomplete. When the endpoint appears healthy but ZDX data does not arrive, service connectivity and allowlisting become part of the investigation.

Authentication, provisioning, and RBAC are also dependencies. ZDX uses user, group, department, location, and device context to scope monitoring and present dashboards. RBAC determines what an administrator can view, modify, export, or run during triage, so least-privilege design should be reflected in operational runbooks.

ZCC health determines whether the endpoint can be trusted as a telemetry source. If ZCC is unauthenticated, ZDX is disabled, the service version is incompatible, or the user is outside monitoring scope, missing telemetry does not prove that the application is healthy. It proves that the observation point is not reliable enough for that conclusion.

Automation should extend validated workflows rather than replace them. The ZDX API can support reporting, alert retrieval, inventory, and troubleshooting workflows, but programmatic access is only as useful as the telemetry and configuration underneath it. The durable model is simple: ZCC is the endpoint sensor, the ZDX cloud is the policy, ingestion, and analytics layer, and Experience Center is the operational view.

🔄 Cross-policy resources

ZDX, ZIA, ZPA, identity, and Client Connector

ZDX monitoring is shaped by adjacent services. ZIA locations, ZPA access paths, identity groups, departments, Client Connector enrollment, service entitlement, and RBAC all influence what data is collected and who can act on it. Strong ZDXA candidates treat these as dependencies in the design, not as trivia.

📝 Quick Review

- ZCC is the endpoint-side telemetry source for ZDX.
- Web Probes measure web application availability and responsiveness, including Page Fetch Time, DNS Resolve Time, Server Response Time, and response-code-based availability.
- Cloud Path Probes measure hop count, latency, packet loss, and path detail between the user and the destination.
- Device health and device events help determine whether the endpoint itself is driving poor experience.
- The Telemetry and Policy Gateway supports the exchange of monitoring metrics and policy between ZCC and the ZDX cloud.
- ZDX Central Authority supports configuration and policy coordination.
- ZDX Analytics aggregates telemetry into Experience Center dashboards and operational views.
- A low ZDX Score is a triage signal, not a root-cause statement.
- Validate ZCC service state, authentication, entitlement, version compatibility, and probe scope when telemetry is missing or stale.
- Use the Performance Dashboard for scope, Applications views for application impact, User Details for user-specific evidence, Cloud Path for path isolation, and device health for endpoint validation.
- Common mistakes to avoid:
- Treating one low user score as proof of an application outage.
- Ignoring ZCC health when telemetry is absent.
- Assigning probes too broadly without a user-population strategy.
- Confusing Web Probe evidence with Cloud Path evidence.
- Escalating intermediate-hop packet loss without checking destination impact and related experience metrics.
- Overlooking RBAC limitations during triage and diagnostics.
- Using API or automation output without validating the telemetry source.

🏅 Exam Blueprint Alignment

Scenario Review

- Section 1.2: Interpret Performance Dashboard evidence and ZDX Scores to determine whether an issue affects a single user, site, region, or wider population.
- Section 1.3: Understand how ZCC interacts with the ZDX cloud.
- Section 1.5: Analyze device health statistics such as CPU, memory, Wi-Fi, disk, and network indicators to determine whether poor experience is endpoint-driven.
- Section 1.8: Analyze last-mile or intermediate ISP performance and identify network anomalies using Cloud Path and related telemetry.
- Section 1.9: Understand the value of the ZDX API for programmatic access to ZDX features after telemetry is validated.
- Section 2.2: Differentiate Web Probes and Cloud Path Probes and select the probe evidence that matches the scenario.
- Section 3.1: Analyze device, network, and application telemetry together to isolate the performance tier causing the issue.
- Section 3.3: Analyze network path information to locate latency or packet-loss bottlenecks across the end-to-end path.

Application View Analysis

Application View Analysis teaches candidates how to use the application perspective as a control point for monitoring and triage. The Applications Overview and application details views help show how monitored applications affect the organization's digital experience, which applications are trending poorly, and which user populations are most affected. This is the right starting point when the scenario gives an application name and asks what to check next.

The blueprint alignment is straightforward: candidates must be able to use the Application view to compare application performance and support broader dashboard interpretation. In practice, this means correlating ZDX Score, Page Fetch Time, affected regions, impacted departments, Zscaler Locations, Web Probe metrics, and Cloud Path or Network Probe metrics. A single metric may start the investigation; correlated evidence should drive the conclusion.

An engineer-level administrator treats the Application view as a scoping surface, not a verdict. If only one user is impacted, pivot to user details. If one region is impacted, narrow by geography and inspect path evidence. If several unrelated applications degrade at the same time, look for a shared location, ISP, Service Edge path, DNS, or endpoint pattern. The dashboard gives you the map; you still have to drive without putting the car in a ditch.

💬 Sidebar

Application view as a decision boundary

The Application view helps answer three operational questions before deeper troubleshooting begins: which application is affected, which population is affected, and which metric family explains the degradation. That boundary prevents premature escalation and keeps the next investigation step tied to evidence.

Comparing Application Performance

Comparing application performance starts with a consistent time range. The selected period applies to the application data being reviewed, so comparing one application over the previous 30 minutes with another over a custom multi-hour window can lead to the wrong conclusion. In exam scenarios, assume the comparison is only meaningful when the time range and filters are aligned.

The first comparison is score movement. Review whether one application's ZDX Score dropped while other applications stayed stable, or whether several applications changed at the same time. A single-application drop suggests an application-specific or application-path issue. A shared drop across unrelated applications suggests a common dependency such as user location, network path, last mile ISP, DNS, Service Edge connectivity, or endpoint conditions.

The second comparison is score movement against Page Fetch Time. Page Fetch Time tracks how long the selected application takes to transfer the fetched page to users, and it is one of the strongest indicators of web application experience. If the ZDX Score drops at the same time

Page Fetch Time rises, the evidence supports a web responsiveness problem rather than a generic “application is bad” conclusion.

The 95th percentile line on the Page Fetch Time chart adds outlier context to that comparison. The average trend shows the central experience across the selected population, while P95 shows the point at which 95 percent of page fetch times are at or below that value. If both the average and P95 rise, the degradation is likely affecting the broader population. If the average remains stable but P95 rises, investigate the impacted users, locations, or regions before dismissing the issue as noise.

Use the additional-application comparison capability when deciding whether a degradation is isolated or shared. Comparing the selected application with other business-critical applications can reveal whether performance changed across a broader delivery path. If Salesforce, ServiceNow, and Outlook all drop for the same region, that is a different problem than Salesforce alone dropping globally.

Application comparison also requires geographic interpretation. The Regions by ZDX Score map shows where users of the selected application are located and how those regions score. If the degradation is concentrated in one city or country, the next step is to inspect the users and path data associated with that geography rather than treating the application as universally degraded.

Departments and Zscaler Locations provide another useful comparison dimension. A department-specific pattern can point to role-based application usage, device standards, browser differences, or identity dependencies. A location-specific pattern more often points toward local network, Wi-Fi, last mile ISP, routing, or forwarding-path conditions.

Probe status is the evidence that explains score movement. Web Probe metrics help explain web responsiveness and availability through Page Fetch Time, Server Response Time, DNS Time, and Availability. Cloud Path or Network Probe metrics help explain latency, packet loss, hop count, and packet count. Candidates should learn to ask which probe family changed when the score changed.

🎓 Exam Note

Same score, different root cause

- Two applications can have similar low scores for completely different reasons. One may show high Server Response Time, while another may show packet loss on the path. The score tells you where to look; the probe evidence tells you what kind of problem you are looking at.
- The administrator should not compare applications only by absolute score. Some applications naturally have different performance profiles because of application design, authentication flow, geography, page complexity, or service architecture. A meaningful comparison looks for deviation, affected users, category change, correlated metric movement, and scope.
- When multiple applications show degradation, compare applications that do not share the same provider or service family. If only one SaaS family is degraded, the provider path or application family may be involved. If unrelated applications degrade for the same population, a shared access path or user-side condition becomes more likely.
- A good comparison workflow usually moves from broad to narrow. Start with the application score trend, compare one or more other applications, inspect Page Fetch Time and probe metrics, evaluate the affected geography and organizational scope, and then pivot to user-level or path-level evidence. Skipping that sequence is how “slow app” turns into a three-hour bridge call featuring 19 people and no facts.

🎓 Exam Tip

Compare before you escalate

- In a scenario with an application complaint, compare the affected application against at least one unrelated application in the same time window. If both degrade for the same users, avoid escalating directly to the application owner until path, location, DNS, and endpoint evidence are reviewed.
- Application comparison also informs containment. If only one application is degraded for one geography, communicate the scoped impact and continue investigating that location. If many applications are degraded across all regions, escalation should be broader and should preserve the shared timing, score movement, and affected population. The right handoff is specific enough that the receiving team can act without rediscovering the problem from scratch.

> ⚠️ **Warning**
> *Do not compare noise*
>
> Brief score dips without supporting metric movement are weak evidence. Confirm whether Page Fetch Time, Availability, Server Response Time, DNS Time, latency, packet loss, or impacted-user counts changed at the same point before declaring a service degradation.

← Compare ZDX Scores

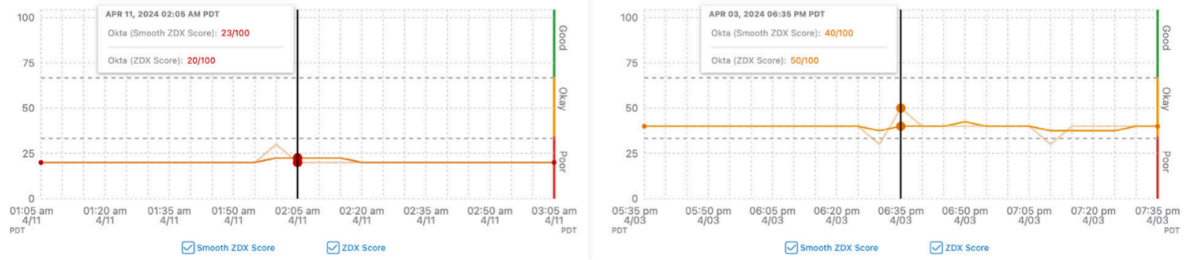
Analyzed Point

User: Giuseppe Conti (giuseppe.conti@thezerotrustexchange.com)
Application: Okta
Time: Apr 11, 2024 02:05 AM PDT

Comparison Point

User: Giuseppe Conti (giuseppe.conti@thezerotrustexchange.com)
Application: Okta
Time: Apr 03, 2024 06:35 PM PDT

ZDX Score

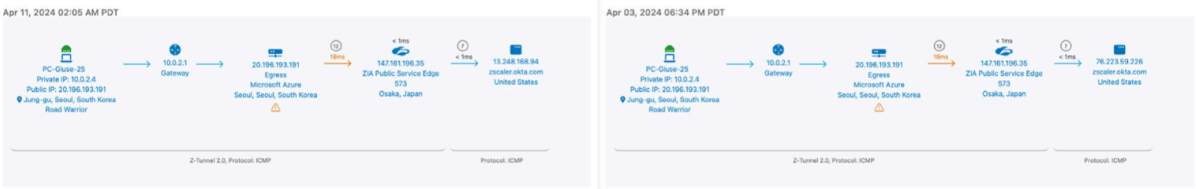


Key Differences

Uptime	152 days, 5:27:16	144 days, 22:01:40
ZDX Score	20	50
DNS Resolution Latency	107.59 ms	60.98 ms
Page Fetch Time	2069.75 ms	1724.49 ms
Application Response Time	1086.88 ms	878.1 ms

Cloud Path

Okta CloudPath Probe — View Detailed Cloud Path



CLOUD PATH METRICS VARIANCE

Destination IP	13.248.168.94	76.223.59.226
Egress-ZIA Public Service Edge Packet Loss	0 %	10 %

Web Metrics

ZDX Score	20	50
DNS Resolution Latency	107.59 ms	60.98 ms
Page Fetch Time	2069.75 ms	1724.49 ms
Application Response Time	1086.88 ms	878.1 ms
DNS to Page Fetch Time Ratio	5.2 %	3.54 %
Application Response Time to Page Fetch Time Ratio	52.51 %	50.92 %

Device Metrics

Uptime	152 days, 5:27:16	144 days, 22:01:40

Identifying Degraded Applications

A degraded application is not simply the application with the lowest number on the page. In ZDX, degradation means the application's experience has become measurably worse for the selected scope and time range, as shown by score trend, affected users, geography, organizational impact, and probe evidence. The candidate should identify degradation by combining these signals rather than reacting to the first red object in view.

The Applications Overview helps identify candidates for deeper review because it presents monitored applications with score trends and impact dimensions. It also helps administrators avoid tunnel vision by showing multiple applications in one place. That matters when the symptom is reported as one application but the evidence shows a broader delivery issue.

Score category is useful but not sufficient. ZDX uses a 0 to 100 score model with Good, Okay, and Poor categories, but the administrator should focus on meaningful movement and business impact. A drop from Good to Poor is obvious. A sustained decline within the Good range may still matter if it affects a critical application used by a sensitive business function.

After selecting the suspect application, correlate ZDX Score with Page Fetch Time. If both move in opposite directions at the same time—score down, fetch time up—the evidence supports degraded user experience for that application. If the score drops but Page Fetch Time does not explain it, examine other probe and path signals before making a root cause claim.

Web Probe Performance helps distinguish slowness from availability problems. Page Fetch Time indicates web fetch duration, Server Response Time reflects the time to first byte, DNS Time reflects name resolution delay, and Availability reflects whether the expected response behavior is observed. The candidate should use these metrics to classify the degradation rather than describing everything as "slow."

Availability changes raise the operational priority. If probes show the application is not available for multiple users or regions, the issue should be handled differently from a mild performance drift. Preserve the affected time range, response behavior, impacted regions, and user count when escalating. The application team needs evidence, not a dashboard screenshot wearing a tiny panic hat.

Network Probe Performance determines whether the application is the source of the degradation or only the visible symptom. End-to-end latency, packet loss, hop count, and packet count help identify whether the path to the application changed at the same time the score degraded. If network metrics explain the score drop, the investigation should move toward the affected path segment.

Geographic concentration is one of the strongest signals in application triage. If the Regions by ZDX Score map shows that only users in a specific city, country, or location group are impacted, the administrator should filter to that population and inspect user and path evidence. A regional outage and a global SaaS issue may look similar to one user, but they should not be handled the same way.

Impacted departments, impacted regions, and impacted locations help convert the application view into an operational scope. Department impact can point toward role-specific usage, user groups, endpoint standards, or identity dependencies. Location impact can point toward local network, Wi-Fi, ISP, or forwarding behavior. Region impact can point toward broader network or provider conditions.

A single-user complaint with a healthy application view should pivot to user-level investigation. In that case, the administrator should examine the user's device, selected application score, Web Probe metrics, Cloud Path data, and device health. This keeps a local problem from being misclassified as an application incident.

Application incidents and application key metrics can provide additional evidence when ZDX identifies a broader issue. Metrics such as application score, HTTP errors, the relationship between server response and page load, last server leg latency, and redirect behavior can help determine whether the condition is application-side, path-side, or mixed. The scenario answer should follow the dominant evidence.

🎓 Exam Tip
Slow, unavailable, or path-impaired

- When identifying a degraded application, classify the symptom precisely. Use Page Fetch Time and Server Response Time for responsiveness, Availability and response behavior for reachability, and latency or packet loss for path impairment. Precise classification leads to the right escalation path.
- Probe configuration can create false degradation if the monitored URL or expected response is wrong. Many applications redirect unauthenticated users to an identity provider. If that redirect is expected and the probe is not designed to evaluate it correctly, a healthy application front door can appear unhealthy. Candidates should recognize that probe design is part of monitoring accuracy.
- Application object design also matters. Grouping unrelated services, regional instances, or multiple application functions into one monitored object can blur the score and weaken troubleshooting. A cleaner design keeps monitored applications aligned to a coherent service boundary so the score represents a meaningful user experience.
- Containment begins by preserving the evidence chain. Capture the application, time range, impacted users, affected locations, score trend, Web Probe metrics, Network Probe metrics, and any user or regional filters applied. A good administrator does not merely escalate; a good administrator packages the problem so the next team can act without starting from square one.

🔄 Cross-policy resources
Evidence handoff

Application degradation can intersect with ZIA, ZPA, Client Connector, identity, and logging workflows. ZDX should identify the experience scope and the most likely failure domain, then the relevant platform evidence should confirm access path, forwarding behavior, authentication dependency, endpoint posture, or service ownership.

SaaS vs Private Application Monitoring

SaaS and private application monitoring use the same operational mindset but different path assumptions. For SaaS, the user experience usually depends on the endpoint, local network, last mile ISP, possible ZIA Service Edge path, internet routing, and the SaaS provider. For private applications, the experience may also depend on the Private Access path, Service Edge, App Connector, and internal application reachability.

Predefined applications simplify SaaS monitoring for common enterprise services because ZDX can provide preconfigured application monitoring behavior. Custom applications require administrators to define the application and probes that represent the service being monitored. The design question is not whether the application is well known; it is whether the monitoring object represents the transaction users actually care about.

For most web applications, Web Probe evidence is central. Page Fetch Time, DNS Time, Server Response Time, and Availability help explain whether users can reach the application and how responsive it is. SaaS troubleshooting should still include Cloud Path evidence because a SaaS symptom can be caused by user-side or network-path conditions rather than the provider itself.

Private application monitoring depends heavily on path interpretation. When users access a private application through Private Access, the relevant path can include the user device, local egress, Service Edge, App Connector, and application. The administrator should determine which segment is impaired instead of treating every private application complaint as an application-server problem.

Probe assignment is especially important for private applications. If an internal application is used only by a specific department or user group, assign monitoring to the relevant users or departments. Monitoring the wrong population creates noise, consumes capacity, and produces conclusions that look scientific but are really just well-formatted confusion.

SaaS and private application comparisons can be powerful when used together. If a user reports that a private application is slow but SaaS applications are healthy, the private application path deserves focus. If SaaS and private applications are both degraded for the same user population, the common elements—endpoint, local network, DNS, ISP, or forwarding path—become more likely.

The Cloud Path view helps separate direct, Internet & SaaS, and Private Access traffic patterns. For SaaS, candidates should consider whether traffic is going direct or through a Service Edge. For private applications, candidates should consider whether the degradation appears before the Service Edge, between the Service Edge and App Connector, or closer to the application. The answer depends on the path evidence, not on the label “private.”

UCaaS monitoring is a special case because meeting quality is not equivalent to loading a static web page. For Microsoft Teams, Zoom, and Webex-style call quality monitoring, ZDX can correlate meeting quality metrics such as latency, jitter, packet loss, and MOS-based scoring with Cloud Path and device evidence. A call quality issue should be evaluated as media experience, not just application availability.

⚠️ Warning

Do not force web-probe logic onto meeting quality

- A normal web application can often be reasoned through Page Fetch Time and Availability. A meeting-quality application requires attention to latency, jitter, packet loss, participant mapping, MOS availability, device state, and media path behavior. The wrong model leads to the wrong team, and then everyone gets to enjoy round two.
- Microsoft Teams call quality data has timing considerations because call records are retrieved after calls complete. Zoom and Webex-style integrations can provide meeting data closer to real time depending on the integration behavior and

prerequisites. The candidate does not need to memorize every integration screen, but should understand that UCaaS evidence may arrive differently from standard synthetic probe evidence.

- Hosted Monitoring provides another vantage point by running probes from Zscaler data centers rather than from user devices. This can help distinguish user-side experience from data-center-to-application performance and availability. It complements endpoint-based monitoring; it does not replace the user-device view that makes ZDX valuable.
- Automation should be used carefully in application monitoring workflows. The ZDX API can support programmatic access to reporting and troubleshooting data, and authorized workflows can use API-based operations such as retrieving application information or initiating approved diagnostics. Use least-privilege roles and clear operational guardrails; “because the script could” is not a governance model.
- The administrator’s escalation path differs between SaaS and private applications. SaaS evidence may lead to the application owner, identity owner, ZIA path owner, ISP, or SaaS provider. Private application evidence may lead to the Private Access owner, App Connector owner, network team, or application team. The best answer in a scenario is usually the one that preserves the path evidence and routes the issue to the team that owns the impaired segment.
- Application View Analysis ends where targeted troubleshooting begins. Once the affected application, population, and likely metric family are clear, the administrator can pivot into User Details, Cloud Path, device health, incidents, diagnostics, or escalation. The point is not to stay in the Application view forever; the point is to leave it with enough evidence that the next move is obvious.

📝 Quick Review

Use a consistent time range and filters before comparing applications.

- Compare ZDX Score trends across related and unrelated applications to determine whether degradation is isolated or shared.
- Correlate ZDX Score movement with Page Fetch Time, Availability, Server Response Time, DNS Time, latency, packet loss, hop count, and affected population.
- Use P95 Page Fetch Time to recognize outlier user experience that averages may hide.
- Use Regions by ZDX Score, impacted departments, impacted regions, and impacted Zscaler Locations to determine scope.
- Treat a low application score as an investigation trigger, not a root cause.
- Separate slow responsiveness, failed availability, and network-path impairment before escalating.
- For SaaS applications, compare against unrelated SaaS applications to identify shared delivery-path issues.
- For private applications, inspect the Private Access path, Service Edge, App Connector, and application-side reachability before assigning cause.
- For UCaaS applications, reason from latency, jitter, packet loss, MOS behavior, device evidence, and Cloud Path, not only web application metrics.
- Use Hosted Monitoring as a complementary vantage point, not a replacement for user-device monitoring.
- Use API-driven workflows with least-privilege access and clear operational guardrails.

- Common mistakes to avoid: escalating before scoping, treating one user as a global outage, ignoring P95 outliers, overlooking expected redirects, mixing unrelated services into one application object, and blaming the SaaS provider before validating user-path evidence.

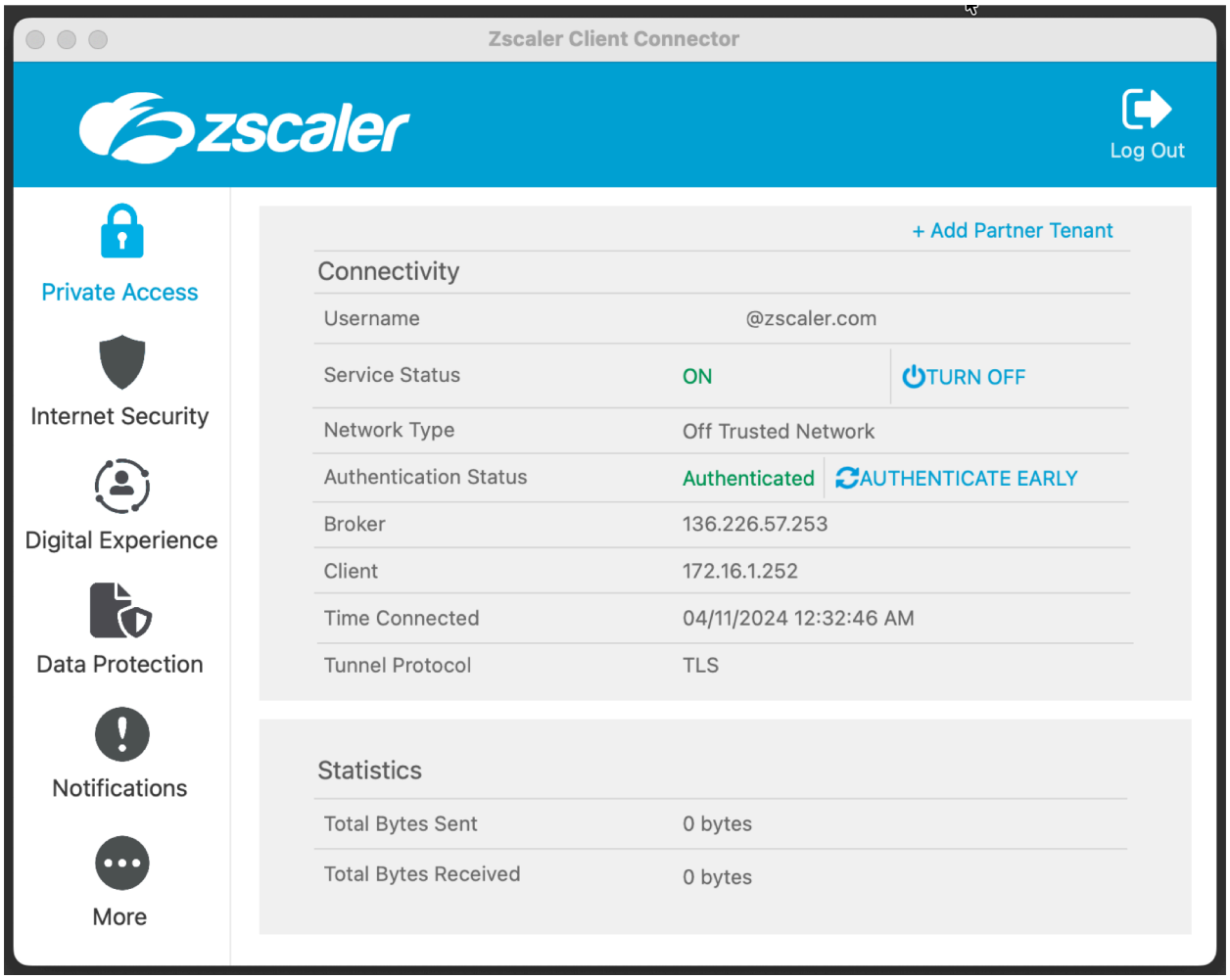
🏅 Exam Blueprint Alignment

Scenario Review

- 1.1 Explain the 0–100 scoring model and use supporting Web Probe and Cloud Path evidence to interpret why the score changed.
- 1.2 Interpret application and performance dashboards to determine whether impact is single-user, site-specific, regional, or widespread.
- 1.4 Use the Application view to compare performance across monitored applications.
- 1.5 Analyze endpoint and local connectivity indicators when application symptoms may be caused by user-device or Wi-Fi conditions.
- 1.7 Understand how third-party integrations such as call quality monitoring support application experience interpretation.
- 2.1 Given requirements, distinguish predefined and custom application monitoring considerations.
- 2.2 Differentiate Web Probes, Cloud Path Probes, and hosted monitoring vantage points when interpreting application behavior.
- 3.1 Analyze application, network, and endpoint telemetry together to identify the most likely degraded tier.
- 3.3 Analyze path information to locate bottlenecks between user, Service Edge, App Connector, and application where applicable.
- 3.4 Use AI-assisted or automated analysis as supporting evidence while still validating the relevant application, path, and device metrics.
- 3.5 Use application and organization-level reporting views to understand recurring performance patterns.
- 3.6 Share scoped evidence through approved snapshot or reporting workflows when escalation requires a view-only current-state artifact.

Device Health and Endpoint Telemetry

Device health and endpoint telemetry describe the condition of the user's device at the time digital experience is measured. Zscaler Client Connector provides the endpoint instrumentation used by ZDX, and Experience Center presents that data alongside application, path, and user experience signals. For ZDXA candidates, the important skill is not memorizing every graph; it is deciding whether endpoint telemetry explains the user's poor experience or simply appears near it.



The administrator baseline teaches the major objects and dashboards. This chapter builds on that baseline by focusing on reasoning: when to trust a device score, when to drill into process-level data, when to pivot into Wi-Fi, and when to widen the scope toward application or network path analysis. The device layer is compelling only when the evidence lines up by user, device, application, and time.

Endpoint telemetry also supports better escalation. A ticket that says "Salesforce is slow" invites guesswork. A ticket that says "the user's score dropped during the selected window while the device showed high CPU and a specific process dominated resource usage" gives desktop

engineering a real starting point. That is the operational difference between monitoring and triage.

💬 Sidebar
Baseline-to-engineer bridge

At the administrator level, know where device and user telemetry appears. At the ZDXA level, know how to use that telemetry to prove or reject an endpoint hypothesis. The exam scenarios are usually testing judgment, not mouse memory.

Device Scoring Dashboard

Use the Device Health dashboard as the practical scoring view for endpoint performance. It consolidates user and device data into a health-oriented view for managed Windows and macOS devices, helping administrators identify poor-performing devices and recognize fleet-level trends. This is where the engineer starts when the question is, “Are bad user experiences being driven by endpoint health rather than application or network behavior?”

The Device Health Score is useful because it summarizes device condition without forcing an administrator to inspect every metric first. The score is based on contributing device metrics such as CPU usage, memory usage, and disk usage. However, the score is not the root cause. It is a prioritization signal that tells you which devices deserve closer inspection.

Score categories make the dashboard easier to operationalize. Good, Okay, and Poor classifications let administrators quickly separate healthy devices from devices that are likely to need attention. In a scenario, a Poor score should trigger investigation of the contributing telemetry, while an Okay score should be treated as context rather than proof of a device problem.

The dashboard becomes more powerful when the engineer filters the population. Department, user group, operating system, geolocation, vendor, model, CPU, memory, and disk views can reveal whether a device-health problem is isolated or patterned. If poor scores cluster around one model or operating system version, the next step is different from a single user with an aging laptop and a heroic number of browser tabs.

Device inventory details help convert a score into an actionable support path. Device name, associated user, operating system, geolocation, vendor, model, CPU, memory, and disk attributes let the administrator determine whether poor health follows a hardware profile, a device cohort, or a specific user. This matters for both immediate troubleshooting and longer-term endpoint lifecycle decisions.

At the user level, device score interpretation should be tied to the selected application and time range. The user’s ZDX Score Over Time shows the experience trend for the application and device, while the device-health section shows whether endpoint conditions changed during that same period. When those timelines align, the endpoint case gets stronger.

Smooth score views can help remove noise from short-term fluctuations, but they should not replace metric-level investigation. Use the smoother trend to understand the direction of the user's experience, then inspect the raw score points and device metrics when explaining a specific event. This prevents a clean-looking trendline from hiding a short but operationally meaningful disruption.

The engineering value of device scoring is proactive action. Repeated poor device health across a model, department, or geography can justify hardware refresh planning, endpoint tuning, patch review, or targeted user support. The goal is not simply to diagnose today's complaint; it is to stop tomorrow's predictable ticket from being born.

🎓 Exam Tip

Read the score, then read the cause

A score tells you where to look. The contributing metric tells you why the score matters. In a scenario, never stop at "Device Health Score is poor" when CPU, memory, disk, Wi-Fi, process, or event evidence is available.

CPU and Memory Utilization Analysis

CPU and memory utilization analysis is the fastest way to test whether a device is struggling to perform the work the user is asking it to do. ZDX exposes endpoint health metrics that help an administrator compare device resource pressure against the user's application experience. When the device is saturated, the user may perceive application slowness even when the web service, network path, and Service Edge path are behaving normally.

High CPU is meaningful when it correlates with the poor-experience window. A spike that appears at the same time as a score drop is worth investigating. A spike that occurs after the user stops working, or long before the reported issue, may be unrelated. The administrator should always align the CPU trend, user score, application selection, and reported time before declaring an endpoint-induced issue.

Process-level evidence is what turns a high CPU observation into an actionable finding. Diagnostic results and detailed device views can expose top processes consuming CPU, memory, disk I/O, or network I/O. A named process with elevated usage during the impacted window gives the desktop, application, or endpoint management team something specific to remediate.

Memory pressure should be interpreted with the same discipline. A device with consistently high memory use may still function acceptably if the user experience remains stable, while a sudden memory spike during a score drop may explain sluggish application response. The useful question is not "is memory high?" but "did memory pressure coincide with the affected application experience?"

Disk and network I/O should not be ignored just because CPU and memory are the familiar villains. A backup agent, software update, sync client, or security process can generate I/O patterns that affect responsiveness without driving CPU to dramatic levels. ZDXA candidates should look for the process and resource combination, not a single headline metric.

Diagnostics sessions provide deeper granularity when dashboard evidence is insufficient. A diagnostics session can collect web probe, Cloud Path, device health, top process, and user device event information based on the selected options. This is appropriate when the issue is intermittent, user-specific, or difficult to prove from normal dashboard sampling alone.

Process Inventory helps move from one-user troubleshooting to operational pattern detection. If a process repeatedly exceeds a configured usage threshold across many devices, the administrator should consider whether a software version, patch, endpoint agent, or background task is creating the experience issue. That is a different escalation path than a one-off user device problem.

Remediation should be scoped to the evidence. A single device with one runaway process may justify user-specific action. A common process across many affected devices may justify software rollback, version review, policy adjustment, or endpoint-management action. Broad remediation without a clear impacted population is how “fixing it” becomes its own outage category.

> ⚠️ **Warning**
> *The endpoint blame button is not a strategy*
>
> High CPU, high memory, and poor score are useful signals, but none of them is root cause by itself. Correlate time, affected population, selected application, device metrics, process evidence, and events before escalating or remediating.

Wi-Fi and Local Connectivity Metrics

Wi-Fi and local connectivity metrics help determine whether the problem is occurring before traffic meaningfully leaves the user's local environment. This is especially important for hybrid and remote users, where the first hop may be a home router, office access point, guest wireless network, or shared workspace infrastructure. ZDX gives the administrator a way to separate local wireless problems from application and broader network issues.

The Wi-Fi Dashboard uses existing user and device Wi-Fi data to monitor performance and identify locations where users may have issues with access points. It can be filtered by SSID, department, Zscaler Location, geolocation, operating system, vendor, and access point prefix. These filters let an administrator determine whether poor experience follows a wireless network, a user population, a device class, or a physical area.

Dashboard summaries help establish whether the issue is broad or localized. Total Access Points, Wi-Fi Performance of Access Points, Total Devices Connected, ZDX Score Distribution, Wi-Fi Band Distribution, and Top Geolocations by Devices Connected provide a quick view of wireless health. If many poor-score devices cluster around the same location or SSID, Wi-Fi becomes a stronger suspect.

Map and list views provide the pivot from region to access point. Wi-Fi Performance by Geolocation can show where devices are experiencing good, okay, or poor performance, and access point views help identify whether one access point is carrying the problematic users. In a scenario, this is the difference between "the office is slow" and "devices behind this access point are having the problem."

Access point details provide the evidence needed for wireless escalation. SSID, BSSID, Wi-Fi performance, device count, source-to-gateway latency, source-to-gateway jitter, and active user information help the engineer describe exactly what is unhealthy. Wireless teams can act on an access point, signal, channel, or roaming pattern much faster than on a vague complaint about "the internet."

The user and device table behind an access point is especially useful. Device, user, BSSID, Wi-Fi adapter, ZDX Score, signal strength, Wi-Fi type, Wi-Fi channel, and operating system can reveal whether the issue is affecting one device or many devices attached to the same wireless infrastructure. A single poor device on an otherwise healthy access point points in one direction; multiple poor devices on the same access point point in another.

Wi-Fi scoring can also be tuned by data collection choices. Depending on configuration, Wi-Fi data collection can use signal strength and retransmission rate to identify low-performing Wi-Fi devices rather than relying only on ZDX Score. That distinction matters when wireless engineering needs radio-quality evidence instead of a general user-experience category.

Local connectivity analysis should still be correlated with web probe and Cloud Path data. If Wi-Fi signal, latency, and jitter look healthy while page fetch time or server response time degrades, the application path deserves more attention. If Wi-Fi metrics are poor while application and Cloud Path signals remain otherwise normal, the access layer is the more credible cause. The best troubleshooting path is rarely linear; it is comparative.

🎓 Exam Note

Wi-Fi privacy and data availability

When Wi-Fi details such as SSID or BSSID are missing, do not assume the environment has no Wi-Fi issue. Validate whether device, operating system, Client Connector, and privacy or collection settings allow that telemetry to be captured.

Identifying Endpoint-Induced Performance Issues

An endpoint-induced performance issue is one where the device or its local access conditions materially explain the user's poor experience. The key word is explain. A device metric that looks imperfect is not enough; the metric must correlate with the affected user, selected application, time range, and symptom. ZDX helps by placing endpoint, Wi-Fi, path, and application evidence in the same operating context.

The strongest endpoint case combines multiple aligned signals. A poor user score during the reported window, elevated CPU or memory, a dominant process, weak Wi-Fi signal, device event, or software crash pattern creates a credible endpoint narrative. The administrator should build the conclusion from converging evidence, not from the first alarming graph that appears on the screen.

Device events are important because they explain instability that ordinary utilization charts may not. System crashes, software crashes, hangs, unexpected shutdowns, and reboot activity can explain intermittent user impact, especially when the user reports symptoms that come and go. When event timing overlaps the poor-experience window, the endpoint hypothesis becomes stronger.

Incident views help determine whether endpoint behavior is isolated or systemic. Device incident categories can show patterns involving system software hangs or crashes, impacted operating systems, impacted software, CPU usage, memory usage, and event trends. That view helps the administrator determine whether the right response is a user-specific remediation or a broader endpoint operations action.

The decision tree should always include a scope check. If only one user and one device show poor experience, endpoint analysis should be deep. If many users in one region show the same application degradation while device metrics are normal, widen the investigation. If many users behind the same SSID or access point are affected, Wi-Fi is more likely than individual hardware failure.

Comparing current performance against a known-good period can reveal what changed. Score comparison, device health trends, Wi-Fi details, Cloud Path metrics, and user device events can show whether the endpoint degraded, the local network changed, or the application/path behavior shifted. This is the cleanest way to turn “it was fine yesterday” into a usable diagnostic statement.

Containment should match the evidence and blast radius. A user-specific runaway process may call for local remediation. A repeated software crash across a device cohort may call for software version review or endpoint-management action. A Wi-Fi issue concentrated on one access point may require wireless engineering. A broad application incident should not be contained by asking every user to reboot, no matter how emotionally satisfying that might feel.

Automation and integration should be governed by least privilege. ZDX APIs, alert delivery, ITSM workflows, and diagnostics access can reduce mean time to resolution, but they should be scoped to the task and role. A service desk administrator may need to view user details or run diagnostics without needing full administrative control. Operational speed is good; operational overpermissioning is how trouble gets a badge and a keyboard.

The exam-level mental model is straightforward: start with scope, correlate the time window, validate the suspected layer, and escalate with evidence. Endpoint-induced performance issues are proven when device or local connectivity telemetry explains the user’s poor experience better than application or network-path evidence. When that proof is weak, keep investigating instead of forcing the device layer to confess.

🔄 Cross-policy resources

Where endpoint evidence goes next

Endpoint findings often hand off to Client Connector administration, endpoint management, wireless engineering, application owners, Service Desk, or ITSM automation. Keep ZDX as the common evidence model, but route remediation to the team that owns the failing layer.

📝 Quick Review

- Start triage by identifying scope: one user, one device, one access point, one location, one region, one application, or many users across the organization.
- Use Device Health Score and ZDX Score as prioritization signals, then inspect the contributing metrics.
- Correlate CPU, memory, disk, network I/O, Wi-Fi, process, and device event data with the same time window as the reported user impact.
- Use process-level evidence to make CPU and memory findings actionable.
- Use Wi-Fi Dashboard data to determine whether poor experience follows an SSID, access point, geolocation, device type, or user population.
- Treat missing Wi-Fi details as a data-collection question before assuming the wireless layer is healthy.
- Use diagnostics sessions when standard dashboard evidence does not provide enough granularity.
- Avoid common mistakes: blaming the endpoint from one metric, ignoring affected-user scope, skipping time correlation, over-remediating without blast-radius evidence, or escalating without process, device, event, and time-window details.

🏅 Exam Blueprint Alignment

Scenario Review

- Explain how score categories support triage while still requiring metric-level validation.
- Interpret Performance Dashboard, user, application, and device views to determine whether impact is isolated or widespread.
- Understand how Zscaler Client Connector contributes endpoint telemetry used by ZDX.
- Analyze device health statistics such as CPU, memory, disk, Wi-Fi, and device events to determine whether poor experience is endpoint-induced.
- Compare current performance against historical or known-good behavior to determine what changed.
- Distinguish endpoint, Wi-Fi, Last Mile ISP, Intermediate ISP, Service Edge, ZPA, and application incident evidence.
- Analyze endpoint telemetry, including resource spikes, Wi-Fi signal strength, software crashes, hangs, and device inventory, to rule in or rule out hardware and local connectivity issues.
- Use diagnostics output to identify a process, device condition, or local connectivity pattern contributing to poor user experience.
- Apply least-privilege thinking when using ZDX roles, diagnostics access, alerting, integrations, APIs, and operational workflows.

Using ZDX Copilot for Baseline Comparison

Baseline comparison is how administrators turn a user complaint into measurable evidence. Instead of asking whether an application “feels slow,” the better question is what changed from normal behavior for that user, application, location, and time window. ZDX Copilot helps accelerate that comparison by summarizing current performance against historical telemetry without requiring the administrator to manually build a custom report.

A useful Copilot workflow starts with a scoped question. Include the application, affected user or population, location or department, current time window, and comparison period. For example, asking Copilot to compare SharePoint performance for Finance users in London over the last four hours against the same period yesterday is far more useful than asking why SharePoint is slow.

The comparison should begin with the ZDX Score trend, then move into the supporting evidence. If Page Fetch Time or Server Response Time changed while Cloud Path stayed stable, the application path becomes more likely. If Cloud Path latency or packet loss changed, the network path needs attention. If CPU, memory, Wi-Fi, or device events changed at the same time, endpoint health must be considered before escalating elsewhere.

Copilot’s value is speed and synthesis, not final authority. Use it to identify the likely delta, affected scope, and next evidence view, then validate the finding in ZDX dashboards, User Details, Cloud Path, incidents, or diagnostics. In ZDXA scenarios, the expected reasoning is simple: ask a precise baseline question, confirm what changed, identify the dominant evidence category, and route the issue to the correct operational owner.

Copilot Overview

ZDX Copilot is an AI/ML-powered conversational assistant that complements ZDX by helping administrators retrieve data, summarize complex telemetry, recognize trends, and reason through possible causes of digital experience issues. It can return relevant insights, graphs, chart visualizations, and concise summaries of larger datasets.

Copilot can interact with many ZDX feature areas, including analytics, troubleshooting, configuration guidance, learning, and optimization. In practice, this means an administrator can ask about user scores, Page Fetch Time, Wi-Fi quality, incidents, alerts, affected applications, or recommended next steps, then use the response to focus the investigation.

Access to Copilot is governed by entitlement and role permissions. Administrators need the appropriate ZDX subscription level and the required permission level for Copilot. In a scenario, that matters because the correct answer might not be “ask Copilot”; it might be “use an administrator role with the correct access” or “escalate to an operator who has the required permission.”

Copilot retains prompts and responses in session history for a rolling 14-day period. That history can be useful when an investigation continues across shifts, but current telemetry still matters. A

clean handoff should include the user or population, application, time window, location scope, and the specific ZDX evidence already checked.

🎓 Exam Note

Copilot is an accelerator

Copilot can provide helpful summaries and recommendations, but its output can be incomplete or inaccurate. Treat every Copilot conclusion as a hypothesis until ZDX telemetry confirms it.

Comparing Current vs Historical Performance

Comparing current performance with historical performance turns a subjective complaint into an evidence-based delta. “The app is slow” is not an engineering conclusion. “Page Fetch Time increased for Finance users in London compared with last known good behavior while Cloud Path latency stayed flat” is something an administrator can act on.

The ZDX Score provides the starting signal for that comparison. It uses a 0–100 scale where higher values indicate better experience and lower values indicate worse experience, with Good, Okay, and Poor score ranges used throughout ZDX dashboards. The score is useful because it compresses multiple user-experience signals into a triage-friendly view, but it must be interpreted with filters, time range, application, and user scope in mind.

A strong comparison workflow begins with a meaningful reference point. ZDX supports comparison against the last known good score and other historical points, such as prior time periods. The comparison view separates the analyzed point from the comparison point and highlights differences across score, metrics, and Cloud Path evidence. That side-by-side view is often the fastest way to determine what changed.

When using Copilot for baseline comparison, define the comparison precisely. Include the application, user or user population, location or department, time window, and historical reference. A prompt that asks Copilot to compare Salesforce performance for users in a specific region during the last four hours against the same period yesterday is much stronger than asking why Salesforce is slow.

Absolute scores can mislead when users are globally distributed. A location that normally operates with higher latency might still be healthy for that geography, while a normally excellent location might require attention after a smaller score change. Engineer-level triage looks for meaningful deviation, not just red, amber, and green paint on the dashboard.

Web probe metrics should be inspected early in a comparison. Page Fetch Time represents the fetch time for the top-level page document, DNS Resolve Time measures name resolution time, Server Response Time reflects time to first byte, and availability reflects whether the configured response behavior was successful. A current spike in Server Response Time with flat network

latency often points more toward the application or application host than toward the access path.

Cloud Path metrics provide the network-side comparison. Latency, packet loss, hop count, and path-leg behavior help determine whether the current issue appears between the device and egress, across the last mile or intermediate ISP path, through a Service Edge segment, or closer to the application. For ZIA and ZPA paths, the Service Edge and App Connector portions of the path become important boundaries for escalation.

Device health comparison keeps the investigation honest. CPU saturation, memory pressure, disk usage, network I/O bursts, Wi-Fi signal changes, battery state, adapter changes, tunnel events, and other device events can all explain poor user experience. These indicators are most meaningful when they align with the same timestamp as the score drop.

Comparing multiple applications helps distinguish a single-service problem from a shared path problem. If only one application degrades for a user population, the application or its hosting path becomes more suspicious. If multiple unrelated applications degrade for the same users or site, the investigation should pivot toward local network, DNS, ISP, Service Edge path, or endpoint conditions.

Performance Dashboard and Applications Overview are the better starting points for population-level comparison. Filtering by application, department, geolocation, Zscaler Location, location group, user group, or last mile ISP helps determine whether the issue belongs to one user population or a broader failure domain. Copilot can summarize this scope, but the filter logic still belongs to the administrator.

User Details is the better starting point for single-user comparison. It provides the selected user, selected device, application score trend, web probe metrics, Cloud Path metrics, device health, and device events in one workflow. Smooth ZDX Score can make the trend easier to read, while point-in-time score inspection helps confirm the specific moment of impact.

The comparison workflow should end with an operational decision. Possible outcomes include monitoring because the variation is within normal behavior, collecting deeper diagnostics, escalating to desktop support, engaging the network team, notifying the application owner, or preparing evidence for Support. A comparison that does not drive a decision is just dashboard tourism.

🎓 Exam Tip

Ask for the delta

When using Copilot, ask what changed between current and historical performance. The most useful output identifies the affected scope, the metric that changed most, the likely evidence category, and the next ZDX view or diagnostic action to validate.

> ⚠ **Warning**
> *Baseline comparison traps*
>
> Avoid comparing against an arbitrary “good-looking” period, ignoring time-zone usage patterns, or using an organization-wide score when the complaint is site-specific. Bad baselines create confident but wrong conclusions, which are still wrong—just wearing a tie.

Using Natural Language Queries for Analysis

Natural language queries are most effective when they are scoped like an escalation note. Copilot needs the same context an analyst needs: who is affected, which application is involved, where the users are, when the issue occurred, and what historical period should be used for comparison.

A weak prompt asks, “Why is the application slow?” A stronger prompt asks Copilot to compare a named application for a defined location and time range against a known-good period, then summarize whether the largest change appears in web probe metrics, Cloud Path metrics, or device health. The second version creates a structured investigation instead of inviting a generic troubleshooting essay.

For user-focused analysis, ask Copilot for a specific user, application, device if relevant, and time range. This style works well for retrieving a user’s ZDX Score, showing Page Fetch Time trends, analyzing Wi-Fi quality, or summarizing whether device health changed during a complaint window. The output should guide the next evidence check in User Details.

For incident-focused analysis, ask about the incident category, impacted users, impacted devices, affected geolocations, and time window. ZDX incidents can represent multi-user issues across categories such as Device, Wi-Fi, Last Mile ISP, ZIA Public Service Edge, ZPA, or Application. The key is to establish whether the incident is a shared operational event or a cluster of unrelated symptoms.

For network analysis, phrase the query around the suspected path segment. Ask whether the current degradation is concentrated in the last mile, an intermediate ISP path, the Service Edge segment, or the path from the Service Edge toward the application. When Copilot summarizes Cloud Path data, validate it with Hop View or Command Line View before assigning ownership.

For Wi-Fi analysis, ask for signal quality, access point latency, SSID patterns, user concentration, and time correlation. A Copilot response that points toward Wi-Fi instability should lead to validation against Wi-Fi signal strength, gateway latency, access point behavior, and related device events. Do not let a local wireless problem masquerade as a SaaS outage.

For collaboration and call quality analysis, natural language can help summarize meeting experience where ZDX has the relevant integration and data. The administrator should look for latency, jitter, packet loss, MOS-related quality indicators, participant impact, and device or

network context. The useful question is not simply whether the meeting was bad; it is whether the evidence points to endpoint, local network, internet path, or service-side conditions.

Copilot can also answer configuration-oriented questions, such as how to configure a webhook, add a ZDX administrator, or configure a probe. In an exam scenario, do not confuse guidance with authorization. If a user lacks the necessary role permission, the correct operational path is to use the right role, follow change control, or involve an authorized administrator.

Optimization queries should be framed around measurable outcomes. Ask Copilot to identify users below a defined score threshold, summarize common factors across those users, or propose an alerting approach that would detect similar degradation earlier. This is where Copilot can help move ZDX from reactive troubleshooting to proactive operations.

Follow-up questions are where Copilot becomes most useful. If Copilot suggests an ISP issue, ask which users, which last mile ISP, which path leg changed, and whether unrelated applications show the same pattern. If Copilot suggests device health, ask which device metric changed at the impacted timestamp and whether the same user had a prior good comparison point.

Prompt hygiene matters. Ask Copilot to separate findings by application, network, and device evidence; request confidence or supporting metric changes; and ask for the next validation step. Avoid prompts that ask Copilot to “fix” the issue without defining the evidence boundary, permissions, or operational guardrails.

🔄 Cross-policy resources

Integrations and evidence flow

Copilot findings often feed other operational workflows: alerts, webhooks, ServiceNow incidents, Diagnostics sessions, snapshots, or API-based reporting. Keep ZDX evidence central. Automation should route, enrich, and preserve evidence; it should not blindly remediate from an unvalidated summary.

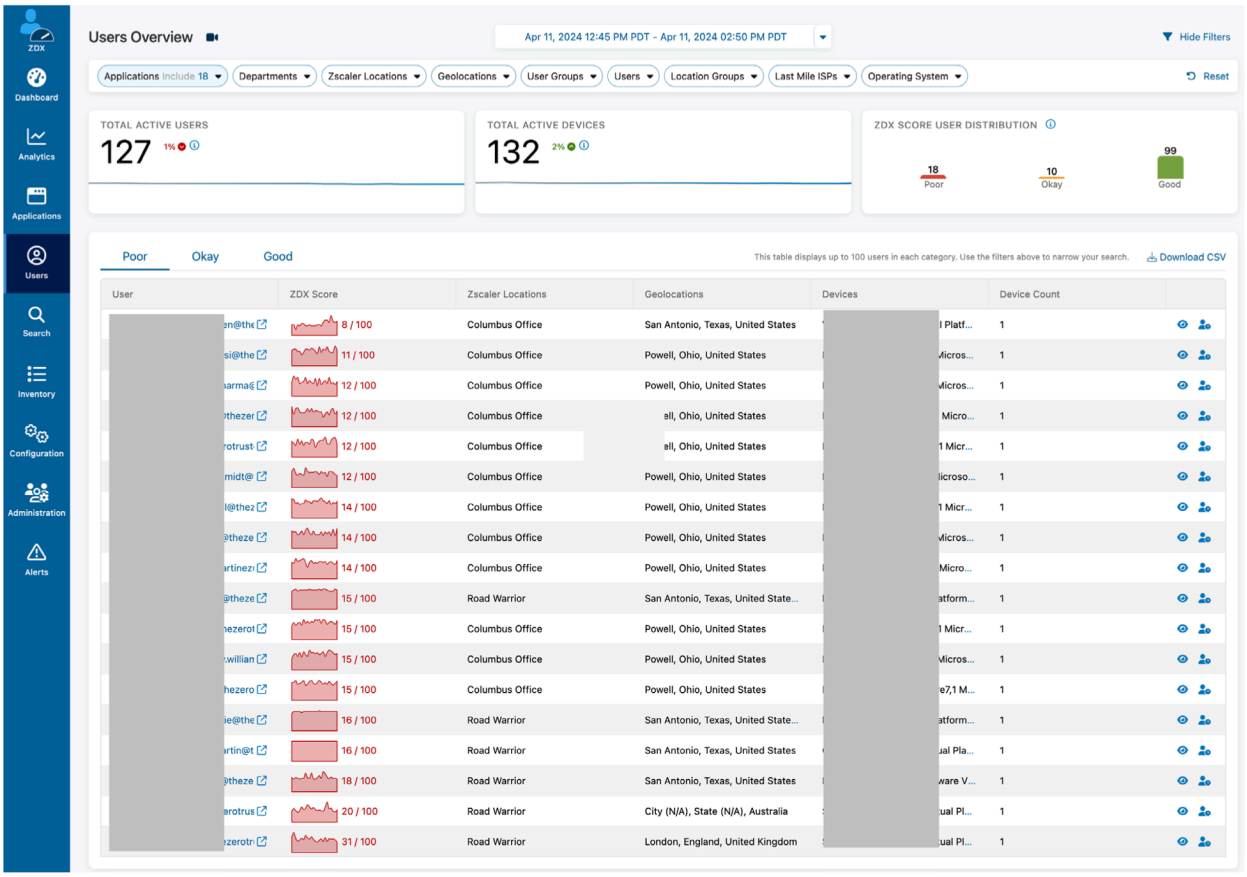
Validating Copilot Findings

Validating Copilot findings is mandatory operational hygiene. Copilot can summarize and recommend, but the administrator must still verify the conclusion against ZDX telemetry. The exam pattern is simple: AI may accelerate the investigation, but evidence closes it.

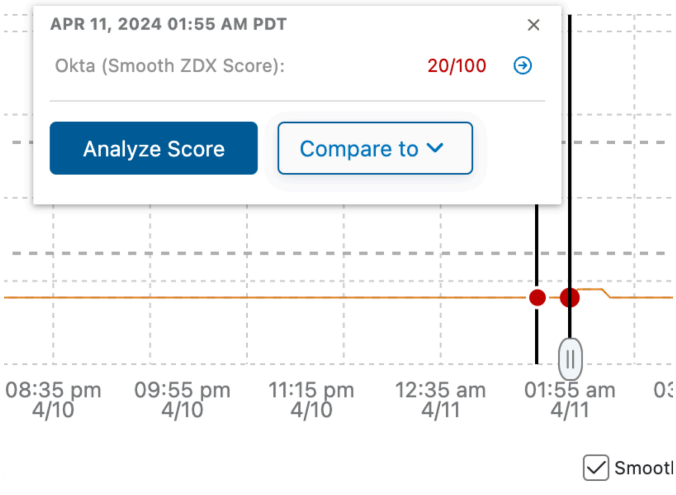
Start by validating scope. If Copilot reports a regional issue, confirm the affected geolocations, Zscaler Locations, users, devices, and applications in the relevant dashboards. If the dashboard evidence does not match the claimed scope, the finding is not ready for escalation.

Next, validate the metric category. Application-related findings should be checked against Page Fetch Time, Server Response Time, DNS Resolve Time, availability, and probe status. Network-related findings should be checked against Cloud Path latency, packet loss, hop count,

and path-leg behavior. Device-related findings should be checked against CPU, memory, disk, network I/O, Wi-Fi, battery, and device events.



Automated root cause analysis can provide a probable factor, explanation, and confidence level for a poor score. The confidence level is helpful, but it is not a permission slip to stop thinking. Confirm whether the proposed factor aligns with the affected timestamp and whether the surrounding graphs support the same conclusion.



Intermediate-hop packet loss requires careful interpretation. Internet devices may rate limit probe traffic, so packet loss at an intermediate hop is less meaningful if later hops and the destination do not show continuing loss. Stronger evidence comes from persistent loss, destination impact, latency trend, and correlation with user-facing web metrics.

Endpoint validation must be time-aligned. High CPU, memory pressure, disk saturation, Wi-Fi signal degradation, adapter changes, tunnel events, or device restarts are most useful when

they occur during the same window as the score drop. If the event happened outside the complaint window, it may be unrelated background noise.

Incident validation should happen before broad communication. If Copilot points to an application, ISP, Wi-Fi, ZIA Public Service Edge, ZPA, or device incident, confirm the incident type, impacted users, impacted devices, location pattern, and persistence over time. A transient single-user blip should not become a global outage notification unless the goal is to invite every manager to a meeting nobody needed.

Diagnostics and Deep Tracing are appropriate when standard dashboard evidence is not enough. A Diagnostics session can collect deeper Web Probe, Cloud Path, device health, top process, and user device event data for the selected user, device, and application. For persistent or difficult issues, this creates a stronger evidence package for escalation.

Evidence preservation matters. Snapshots and exported diagnostic results can help support handoffs between Service Desk, desktop, network, application, security, and Support teams. When sharing evidence, preserve what the next team needs while respecting role-based access, obfuscation, and privacy expectations.

Automation and API workflows require the same validation discipline. ZDX API access, webhooks, and ServiceNow integrations can enrich tickets, launch diagnostics, and move evidence into the operational workflow, but they should be designed with least privilege and clear ownership. Automated routing is useful; automated confusion just moves faster.

A validated Copilot workflow closes the loop. The administrator should document the scoped population, the current-versus-historical change, the dominant evidence category, the validation source, and the next action. That is the difference between “AI says it is the network” and “Cloud Path latency increased on the last mile segment for affected users during the complaint window while web and device evidence remained otherwise stable.” One is a guess. The other is engineering.

📝 Quick Review

- Use Copilot to accelerate ZDX analysis, not to replace evidence review.
- Start with scope: single user, site, region, application, ISP, Service Edge path, or widespread issue.
- Compare current behavior against a meaningful historical reference such as last known good behavior or the same business period.
- Separate evidence into application, network path, and device health categories.
- Validate Copilot findings against ZDX dashboards, web probe metrics, Cloud Path, User Details, Incidents, and Diagnostics.
- Treat intermediate-hop packet loss carefully; confirm whether loss persists to later hops or the destination.
- Use precise prompts that include application, user or population, location, time range,

and comparison period.
- Preserve escalation evidence with snapshots, diagnostics, exports, or ticket enrichment where appropriate.
- Avoid common mistakes: trusting Copilot without validation, comparing against the wrong baseline, ignoring endpoint health, over-escalating transient issues, and routing tickets before confirming scope.

🏅 Exam Blueprint Alignment

Scenario Review

- Explain how the 0–100 ZDX Score supports monitoring and triage decisions.
- Interpret Performance Dashboard and application scores to determine whether degradation affects a single user, site, region, or broader population.
- Use Application view to compare different applications and separate application-specific degradation from shared-path degradation.
- Analyze device health statistics to determine whether CPU, memory, Wi-Fi, disk, network I/O, or device events contribute to poor experience.
- Use Copilot to compare current performance against historical baselines without manually building custom reports.
- Use AI-assisted analysis to accelerate root cause investigation while validating the finding against ZDX telemetry.
- Analyze Last Mile and intermediate ISP performance by correlating Copilot summaries with Cloud Path evidence.
- Understand how Call Quality Monitoring, ServiceNow, alerts, webhooks, snapshots, Diagnostics, and API workflows support triage, escalation, and operational follow-through.

Third-Party Integrations and Ecosystem

The Third-Party Integrations and Ecosystem objective sits inside Monitoring and Triaging Digital Experience because modern operations rarely happen in a single console. UCaaS teams use call-quality systems, service desks work from ServiceNow, endpoint teams rely on Microsoft Intune, and automation teams consume APIs. ZDX connects these workflows to the same experience evidence used for monitoring and troubleshooting.

The ecosystem can be understood as four integration patterns. Call Quality Monitoring enriches ZDX with meeting and media-quality telemetry. ServiceNow operationalizes ZDX alerts and diagnostics inside incident management. Microsoft Intune contributes Endpoint Analytics for device health and reliability. OneAPI and ZDX API access allow external systems to retrieve, report, and act on ZDX data programmatically.

Integration design begins with a practical question: what system initiates the workflow? If ZDX detects an alert, a webhook or ITSM integration may be the right mechanism. If a dashboard needs historical scoring, inventory, or alert details, an API pull model may be better. If a service desk analyst needs to collect deeper evidence during a ticket, ServiceNow-driven Deep Tracing can reduce context switching.

Good operational design also asks who can act on the evidence. A help desk analyst may need to see impacted users and run a bounded diagnostic session, while an automation service may only need alert details and affected devices. Role design, API key management, monitored-user scope, and ticket routing all need to match the operational action being delegated.

🔄 Cross-policy resources

Ecosystem boundaries

ZDX integrations intersect with Client Connector, identity, ZIA, ZPA, alerting, administration, and automation. Keep the primary diagnostic question anchored in ZDX: who is impacted, what changed, when did it change, and which layer owns remediation? Use external systems to enrich, route, or automate the workflow only where they materially improve that decision.

Call Quality Monitoring (Teams, Zoom)

Call Quality Monitoring is designed to analyze actual calls and meetings, not just whether a collaboration application login page is reachable. Microsoft Teams Call Quality and Zoom Call Quality are UCaaS monitoring applications that evaluate media experience using call-quality metrics. This is different from monitoring the Microsoft Teams Web App or Zoom Web App as web applications.

For Microsoft Teams, ZDX retrieves call-quality data through the Microsoft Graph API after the call has completed. The Microsoft Graph API provides data for the entire call after completion rather than incremental call data while the meeting is still in progress. In a triage scenario, this means Teams call-quality evidence is primarily post-call diagnostic evidence.

Zoom behaves differently. Zoom Call Quality data is retrieved using the Zoom API, and data can be captured in real time as a meeting is in progress. Complete call data becomes available after the call ends. ZDX supports onboarding either a Zoom API tenant or a Zoom QSS tenant, and the design choice should reflect the organization's scale, API usage, and need for pushed call-quality data.

Autosense Cloud Path probes are important because UCaaS media destinations can change from meeting to meeting. A regular Cloud Path probe to a fixed host might not represent the actual media relay used by a call. When a Teams or Zoom Call Quality tenant is onboarded, ZDX can use Autosense Cloud Path behavior to detect the destination IP address and correlate media quality with the network path actually used by the session.

The Call Quality score can be calculated from MOS or from metric thresholds. When latency, jitter, and packet loss are all available, MOS can be used to determine the Call Quality ZDX Score category. MOS values from 0 through 3.6 map to Poor, values greater than 3.6 and less than 4.34 map to Okay, and values of 4.34 or greater map to Good. When only one or two call metrics are available, metric thresholds can be used instead; latency above 500 ms, jitter above 30 ms, or packet loss above 0.1 percent can drive a Poor determination. If latency, jitter, and packet loss are all unavailable, the Call Quality ZDX Score cannot be calculated.

🎓 Exam Note

Call-quality evidence

- A blank MOS or missing ZDX Score is not proof that a meeting was healthy. It usually means the required data was unavailable, incomplete, or could not be mapped. In scenario work, missing data should narrow your confidence, not widen your assumptions.
- In triage, begin with the impacted meeting and the affected participants. Then ask whether the poor experience is concentrated in a location, Zscaler location, department, user group, host, or individual device. A meeting with one degraded participant is a very different scenario from a meeting where multiple users in the same region show elevated jitter or loss.
- Meeting details help connect call quality to endpoint and network evidence. Depending on what is available, ZDX can show meeting identifiers, host information, session details, device operating system, IP address, geolocation, audio type, ZDX Score, and MOS. Clicking into an instrumented user gives the administrator a path to compare call quality with User Details, device health, device events, and Cloud Path metrics.
- Identity and participant mapping are common sources of confusion. External users, guest users, users not monitored for Call Quality, or devices that cannot be mapped to registered ZDX devices may not expose the same detail as managed users. Microsoft may also truncate or hide certain participant details for privacy reasons. Treat those conditions as visibility boundaries.
- Configuration scope matters because Call Quality is only as useful as the monitored population. Monitoring criteria should include the users whose meetings need to be observed, and tenant authentication must remain valid. When monitoring criteria are updated, the tenant may need to be reauthenticated so the integration continues to

align with the selected users.
- Do not confuse collaboration application access with call media quality. Page Fetch Time, DNS Time, Server Response Time, and availability are useful for web application monitoring. MOS, latency, jitter, packet loss, meeting participants, and Autosense Cloud Path context are the relevant signals for call-quality triage. The two can coexist, but they answer different questions.
- Call Quality alerting should be scoped carefully. Call Quality alert criteria can include MOS or ZDX Score and meeting or participant impact criteria. A useful alert should represent an operationally meaningful condition, such as multiple impacted participants or repeated degradation, rather than a single noisy data point that creates ticket confetti.

⚠️ Warning
UCaaS timing traps

Teams call-quality data is post-call. Zoom can provide incremental data while a meeting is in progress, but complete data is available after the call ends. External users, guest users, API constraints, unmonitored users, and unmapped devices can all limit visibility. Validate the monitored population before making a strong operational conclusion.

ServiceNow Integration

The ServiceNow integration is designed to reduce swivel-chair triage by bringing ZDX alert and diagnostic context into the incident workflow. Instead of forcing a service desk analyst to manually copy evidence from ZDX into a ticket, the integration can expose ZDX alert impact, impacted users, Deep Tracing, and root cause analysis where incident work is already happening.

Service users and roles are central to the design. The integration supports different ServiceNow roles depending on whether the workflow needs Root Cause Analysis, Alerts, Deep Tracing, or a combination of those capabilities. A service user that only routes alert details should not automatically receive the same functional access as one allowed to start diagnostics.

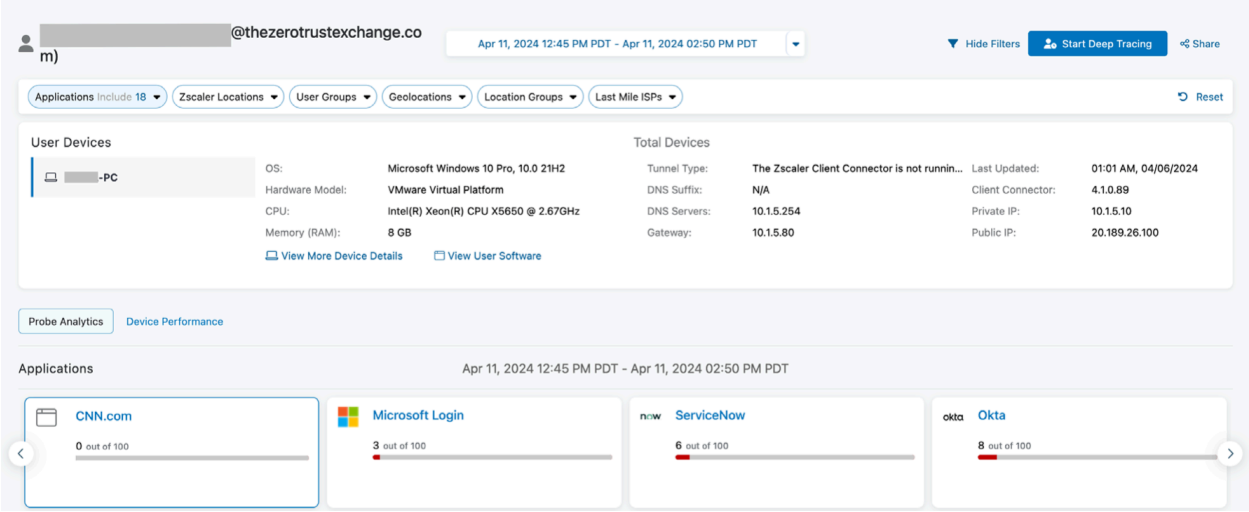
The ServiceNow settings require ZDX connection details such as the ZDX Admin Portal URL, ZDX Public API URL, API Key ID, and API Key Secret. Those values enable ServiceNow to interact with ZDX for functions such as Deep Tracing and root cause analysis. Treat those values as operational credentials, not as documentation filler.

ServiceNow can also receive ZDX alerts through webhook-based workflows. Webhooks can deliver alert information about application, device, or network performance, and ServiceNow can support incident management or event management use cases depending on the implementation. Multiple alert rules can use the same webhook when the downstream workflow is designed to handle them.

Incident mapping is what turns raw alerts into actionable queues. ZDX alert types can be mapped to ServiceNow categories and subcategories so device, network, application, or ZDX Score-driven issues route consistently. Alert Impact and Impacted Users views help the analyst understand the affected departments, geolocations, Zscaler locations, devices, and users.

Deep Tracing from ServiceNow is useful when a ticket needs deeper evidence for a specific user, device, and application. The analyst can start a bounded session, select the impacted user and device, choose the relevant application and probes, and run the session for an approved duration. After completion, ServiceNow can surface session metadata and a ZDX link to review results.

Root Cause Analysis from the ZDX Summary view helps the analyst evaluate why an application score dropped during the incident window. The output can identify likely contributing factors, but it should still be validated against the supporting ZDX evidence. Use it to accelerate investigation, not to skip investigation.



User identity alignment deserves attention. If ServiceNow usernames or email domains differ from the Client Connector login identity, User Domain Settings can append or replace a domain string to improve alignment for ZDX Summary and Deep Tracing operations. This setting supports identity matching; it does not rewrite existing ServiceNow user records.

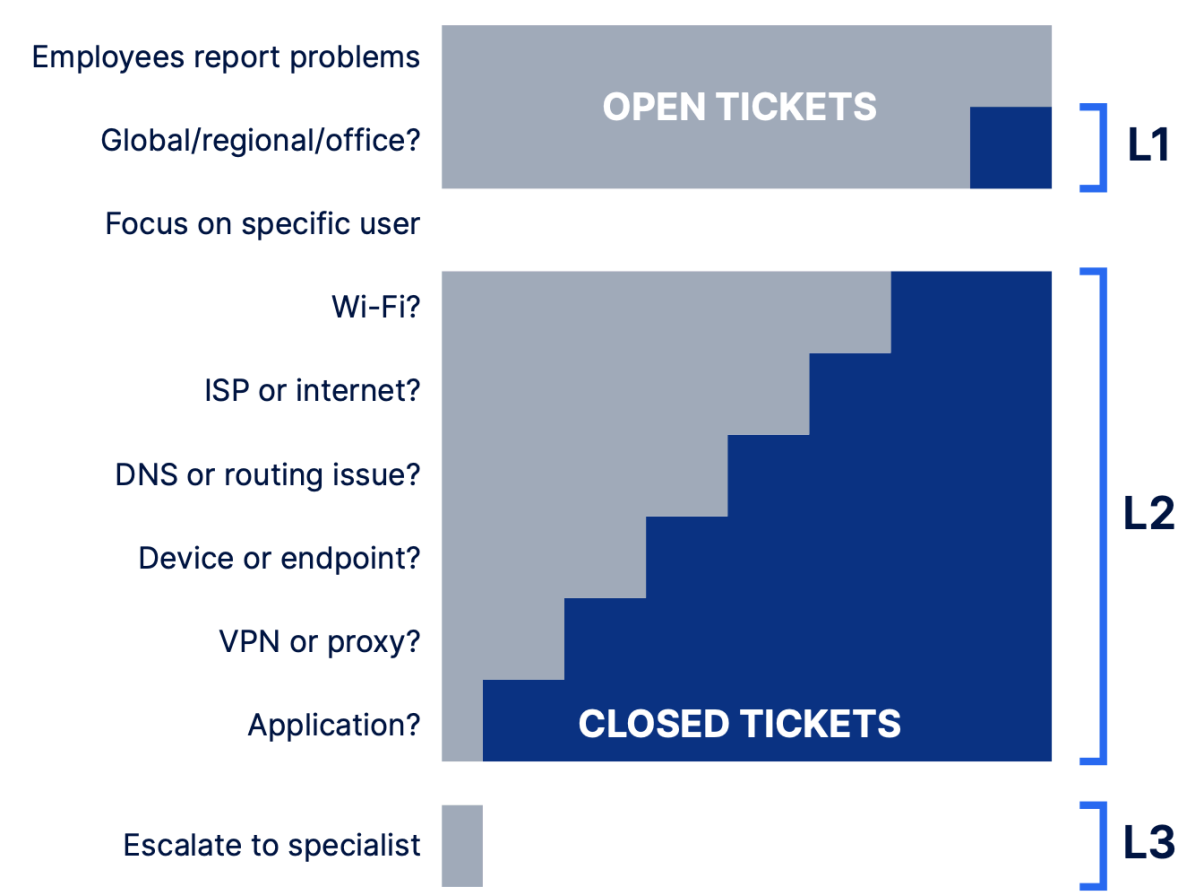
The main implementation trap is automating ticket creation before alert criteria and routing logic are mature. A noisy alert rule can create a noisy ServiceNow queue. Design for meaningful thresholds, impacted-user context, clear assignment groups, and a route back to ZDX evidence so analysts can confirm the actual experience impact.

🎓 Exam Tip

ServiceNow triage

When a scenario asks how to reduce MTTR for service desk analysts, look for the workflow

that keeps ZDX evidence close to the ticket: alert impact, impacted users, Deep Tracing, and root cause analysis. When the scenario asks about governance, focus on service user roles, API credentials, category mapping, and controlled diagnostic permissions.



Microsoft Intune Integration

Microsoft Intune integration adds Endpoint Analytics to the ZDX user and device view. After Intune is integrated with ZDX, metrics from the Microsoft Intune API can be mapped to individual ZDX users and devices. This gives the administrator a stronger basis for deciding whether the root cause is endpoint reliability rather than application or network performance.

Startup Performance is useful for complaints that begin before the user reaches the application. Endpoint Analytics can show a startup score and supporting boot and sign-in history. Core Boot focuses on the time from system power-on to sign-in, while Core Sign-in focuses on the time from credential entry until the rendered desktop is usable and CPU usage has stabilized.

The same view can expose processes that affect startup. When a user complains that “everything is slow,” startup-impacting processes help distinguish endpoint initialization problems from SaaS, Service Edge, or last-mile network problems. That distinction changes the escalation path.

Software Reliability addresses a different failure pattern. Application crashes, software events, and reliability history can explain symptoms that look like application slowness but are actually caused by unstable endpoint software. If an application crashes repeatedly near the complaint window, the administrator should not lead with a network escalation.

Endpoint Analytics should be correlated with native ZDX device telemetry. ZDX already collects device health signals such as CPU, memory, disk, network I/O, Wi-Fi signal strength, and device events. Intune data adds endpoint context; Web Probe and Cloud Path data still explain application reachability and network path behavior.

Monitoring criteria affect what data is visible. Intune integration scope can be constrained by filters such as user groups, departments, users, devices, or related criteria. Selections across different filters narrow the monitored population, while selections within a single filter broaden it. If expected Endpoint Analytics data is missing, confirm that the user and device are actually in scope.

In a scenario, a single user with poor startup performance, crash events, high CPU, and normal Cloud Path behavior is usually an endpoint workflow. A group of users in one city with normal Endpoint Analytics but degraded application score and path latency points elsewhere. The best answer is the one that follows the evidence tier, not the loudest complaint.

A mature administrator uses Intune data to improve triage precision, not to duplicate ZDX. The goal is to decide whether the service desk should remediate software, investigate startup processes, review device health, escalate to endpoint engineering, or return to application and network evidence. Intune is the endpoint narrator; ZDX still ties the story to user experience.

OneAPI and Programmatic Integration Use Cases

OneAPI and the ZDX API provide programmatic access to ZDX features so digital experience evidence can be used outside the ZDX Admin Portal. Common use cases include operational dashboards, SIEM enrichment, ITSM workflows, custom reporting, and targeted automation for troubleshooting. The administrator-level skill is knowing which API pattern fits the operational requirement.



ZDX API capabilities include authentication, administration, alerts, inventory, reports, configuration, and troubleshooting. Authentication provides access tokens for API use. Administration can retrieve active locations and departments. Alerts can retrieve alert details for

device, application, network performance, and ZDX Score. Inventory and reports support software distribution, application scoring, and device health analysis.

Time-window handling is a testable operational detail. Some Alert API and Report API endpoints require a two-hour time range. If a workflow needs four hours of data, it should issue multiple two-hour requests rather than one oversized request. Some alert-specific endpoints, such as retrieving an alert by ID or retrieving affected devices for an alert, do not require the same time-window parameter.

Troubleshooting APIs can be used to start or stop Deep Tracing for a device. This is valuable for controlled diagnostic workflows, but it should not be used as a broad, always-on tracing mechanism. API calls are rate limited, and Deep Tracing is intended for focused evidence collection against an identified user, device, and issue.

Webhooks and API polling solve different problems. Webhooks are appropriate when ZDX needs to send event-driven alert data to another system. API access is appropriate when another system needs to query ZDX for reporting, enrichment, inventory, affected devices, or targeted troubleshooting. Webhooks can use authentication methods such as Basic, Token, or OAuth, and ServiceNow can use OAuth 2.0 in supported webhook configurations.

Access governance matters because programmatic access can expose user, device, application, location, and performance data. API keys, client secrets, bearer tokens, and service accounts should be scoped to the workflow, protected as credentials, and reviewed periodically. Device and user visibility should be granted deliberately, especially for integrations used by broad service desk teams.

Automation should preserve evidence and validate outcomes. A good workflow records which alert triggered, which users or devices were affected, what action was taken, and whether the ZDX Score, alert status, or diagnostic evidence changed afterward. Closing a ticket without checking recovery is just moving the problem from the queue to the user’s blood pressure.

A realistic workflow might retrieve affected devices for a ZDX alert, enrich the incident with inventory or user context, create or update a ServiceNow record, and trigger Deep Tracing only for a selected device that still shows active symptoms. That is materially different from launching diagnostics against every impacted user at once.

Scenario selection is straightforward when you map the requirement to the integration. Use Call Quality Monitoring for Teams or Zoom media experience. Use ServiceNow when the requirement is incident routing or service desk workflow. Use Microsoft Intune when endpoint startup, sign-in, reliability, or crash evidence is needed. Use OneAPI or the ZDX API when the requirement is programmatic access, enrichment, reporting, or controlled automation.

📝 Quick Review

Integrations should enrich ZDX evidence, route incidents, or automate controlled actions; they should not replace ZDX triage.

- Teams Call Quality data is retrieved after the call has completed; Zoom can provide in-progress data, with complete data available after the call ends.
- Call Quality scoring can use MOS when latency, jitter, and packet loss are available; metric thresholds are used when only partial metric data is available.
- Autosense Cloud Path is important for UCaaS because media destinations can change dynamically.
- ServiceNow integration can expose alert impact, impacted users, Deep Tracing, and root cause analysis in the incident workflow.
- Microsoft Intune integration adds Endpoint Analytics for startup performance and software reliability, which helps identify endpoint-driven user experience issues.
- OneAPI and ZDX API use cases include alerts, reports, inventory, administration, authentication, and troubleshooting automation.
- Common mistakes to avoid: treating missing MOS as a healthy call, over-privileging service users, automating noisy tickets, ignoring monitored-user scope, polling beyond API time-window limits, and escalating endpoint evidence to the network team by reflex.

🏅 Exam Blueprint Alignment

Scenario Review

- Section 1.2: Interpret dashboards and scores to determine whether impact is isolated to a user, site, region, application, or broader population.
- Section 1.5: Analyze device health and endpoint telemetry to determine whether user experience is driven by endpoint health or local connectivity.
- Section 1.7: Understand the value and benefits of Call Quality Monitoring, ServiceNow, Microsoft Intune, and OneAPI integrations.
- Section 1.9: Understand the significance of the ZDX API for gaining programmatic access to ZDX features.
- Section 3.2: Analyze diagnostics session output and use Deep Tracing evidence to support root cause determination.
- Section 3.4: Use root cause analysis capabilities to accelerate time to resolution while validating the recommended cause against supporting telemetry.
- Section 4.2: Configure alert rules, trigger notification types, webhook delivery, ServiceNow integration, and OAuth-aware alert workflows.

Network Intelligence and ISP Analysis

Network Intelligence provides a structured way to analyze network behavior from the user perspective. It uses Cloud Path probe data, network latency, packet loss, baseline comparison, and anomaly detection to help show whether a problem is closer to the user's local access path, the Last Mile ISP, an intermediate provider path, the Zero Trust Exchange, or the application side of the connection.

This subsection aligns to scenario-based monitoring because ISP-related questions are almost always about decision quality. When users report that an application is slow in a region, the correct response is not to assume an application outage, blame the Service Edge, or dispatch the help desk into one laptop. The better response is to narrow the evidence by application, affected population, geography, ISP, time window, and path segment.

Use Network Intelligence together with the Performance Dashboard, Applications Overview, Users Overview, Incidents Dashboard, and User Details. The Performance Dashboard helps identify application and regional scope. Network Intelligence helps explain whether the network path is behaving abnormally. User Details and Cloud Path views help validate whether the observed condition follows the affected user, device, ISP, or application.

🎓 Exam Note

Blueprint focus

This chapter primarily supports the objective to analyze Last Mile or Intermediate ISP performance and identify network anomalies. It also reinforces scenario decisions involving Performance Dashboard scope, Application view comparison, endpoint health interpretation, and network path bottleneck analysis.

Last Mile Performance Analysis

Last Mile performance analysis focuses on the user's nearest access segment: the local network and the ISP path that carries traffic away from the user before it enters broader provider, Service Edge, or application-hosting paths. For remote users, this segment can include home broadband, cellular connectivity, hotel networks, shared apartment Wi-Fi, or a regional ISP that the enterprise does not control.

A Last Mile ISP hypothesis becomes stronger when affected users share a provider, a geography, a time window, and a similar experience pattern while other users of the same application remain healthy. It becomes weaker when the same user has device health issues, unstable Wi-Fi, or when other applications through the same ISP are unaffected. In other words, correlation is useful; lazy correlation is still lazy.

Last Mile incidents can present as blackout or brownout patterns. A blackout is primarily connectivity-oriented and is evaluated with evidence such as application score, probe errors, DNS resolution time, and Cloud Path probe errors. A brownout is primarily performance-oriented

and is evaluated with evidence such as ZDX Score drop, DNS latency, leg latency, and packet loss.

When the Performance Dashboard shows degradation in a region, begin by selecting the affected application and narrowing by geography or location. If the population is distributed, Last Mile ISP filtering can reveal whether the issue follows a provider rather than a physical office, city, or department. That distinction changes the next action: a site issue, a provider issue, and an application issue do not belong in the same escalation lane.

Do not confuse Wi-Fi symptoms with Last Mile ISP behavior. Poor signal strength, unstable access point latency, retransmission issues, or a weak local gateway path can make an ISP look guilty when the problem never really leaves the user's local environment. Start by separating local connectivity from provider behavior before escalating outside the organization.

DNS deserves special attention in last-mile analysis because name resolution can affect the perceived application experience even when the application is available. If DNS latency increases while application server response and later path segments remain stable, the problem may be closer to DNS resolver behavior, local network configuration, or provider DNS performance than to the application owner.

Cloud Path evidence helps place the issue along the path. If the affected latency or loss appears early and aligns with users on the same provider, the Last Mile ISP hypothesis becomes more defensible. If the early path is stable and degradation appears closer to the application or Service Edge path, the administrator should shift the investigation accordingly.

The operational handoff for a last-mile issue should include the affected ISP, geolocation, application, time range, number of impacted users, relevant DNS behavior, path segment evidence, and whether the condition appears to be blackout or brownout. That is the difference between an actionable escalation and "the internet is slow," which is technically a sentence but operationally a cry for help.

🎓 Exam Tip

Last-mile decision pattern

For Last Mile ISP scenarios, look for a shared ISP or access geography, matching user-impact timing, relevant DNS or leg latency changes, and Cloud Path evidence that places the degradation near the user side of the path.

⚠️ Warning

Common last-mile trap

Do not classify a problem as Last Mile ISP merely because the user is remote. Remote work adds possible causes; it does not automatically identify the cause.

Intermediate ISP Behavior

Intermediate ISP behavior refers to degradation that occurs after the user's last-mile access path but before the application is reached. This can include issues inside an ISP backbone, between two ASNs, at a peering relationship, or along a provider path toward the Zero Trust Exchange or an application service.

Intermediate ISP incidents may be classified as internal provider behavior, ISP-to-ISP peering behavior, or provider behavior associated with a path toward a Service Edge data center. When probes are routed through a data center, the epicenter can be shown at that data center location. When probes are direct, the epicenter can be closer to the approximate location of affected users. The administrator must interpret the epicenter in the context of the path, not as a standalone blame marker.

The ISP or ASN view is useful because it aggregates path information by provider identity rather than forcing the administrator to read every individual hop. By selecting an ASN node, the administrator can inspect how IP addresses and provider relationships contribute to the observed path. This is especially valuable when multiple users are affected but the issue is not local to their immediate ISP.

Intra ASN and Inter ASN distinctions matter. Intra ASN evidence points to behavior within the same autonomous system. Inter ASN evidence points to behavior between autonomous systems, such as a peering point. A scenario that describes degradation "between providers" or "across a backbone path" should push the candidate toward Inter ASN reasoning rather than a simple last-mile conclusion.

Percentile latency is more instructive than average latency when user experience is uneven. P50 describes the median probe experience for a link, while P90 and P95 expose higher-latency tails. If the average looks acceptable but P95 is elevated during the complaint window, the administrator should not dismiss the issue simply because the mean value looks polite.

Network Intelligence severity is tied to deviation from baseline, with medium and high conditions associated with larger deviations and minimum latency thresholds. If a baseline is not established, default threshold values are used. The key concept is that severity reflects deviation from expected behavior, while the operational priority still depends on user impact.

A highlighted intermediate path does not automatically mean an outage. It means the measured path is abnormal compared with expected behavior. Before escalating, confirm whether poor ZDX Scores, affected applications, impacted users, and complaint timing line up with the anomaly. The internet can be weird and harmless at the same time; that is why baselines and user impact both matter.

A useful intermediate ISP escalation contains the affected ASN or ASN relationship, path direction, time window, baseline deviation, percentile latency, packet loss evidence when

present, impacted users, and affected applications. This allows network operations or provider support teams to investigate the right segment without replaying the whole discovery process.

💬 Sidebar
Reading provider evidence

ASN and peering views identify where measured path behavior changes. They do not, by themselves, prove ownership or fault. Use them to scope the path, support escalation, and avoid confusing application symptoms with provider-path behavior.

Identifying Network Anomalies

A network anomaly is not simply “high latency.” Latency is a normal part of any network path. An anomaly becomes operationally meaningful when network behavior deviates from baseline and aligns with affected users, poor scores, packet loss, application degradation, or a specific time window.

The Network Intelligence overview helps the administrator prioritize before drilling into details. Recent anomaly counts show where network behavior has changed. Top cities by user count help identify where a large user population could be affected. High-latency ASN indicators help surface provider paths whose latency is elevated relative to baseline.

The map view helps locate impacted geographies, but it must be interpreted carefully. Larger geolocations can show highlighted anomaly behavior, while smaller user populations may require selection before anomaly data is populated. Absence of a visible map effect is not always proof that no anomaly exists; it may reflect the way the dashboard populates data for lower-volume locations.

Application filtering changes the investigative question. When an application filter is selected, the map focuses on geolocations where users are accessing that application. In that mode, anomaly data may need to be populated by selecting a geolocation. Candidates should understand that a network-wide view and an application-filtered view can produce different investigative paths.

The User Experience & Network Insights chart connects network conditions to user impact. It can show anomaly timing, probe count, network latency, packet loss percentile, and users with poor ZDX Scores. Probe count helps establish confidence in the measured condition, while the poor-score overlay helps determine whether the anomaly is actually hurting users.

A network anomaly without poor user experience should be handled differently from an anomaly that overlaps with poor ZDX Scores. A path can deviate from baseline without causing meaningful business impact. For example, a latency increase from a very low baseline may be statistically abnormal but still acceptable for the user experience. The administrator should distinguish “changed” from “broken.”

Performance tables help convert a visual anomaly into an operational priority. The ISP Performance Table can surface Last Mile ISPs by baseline deviation, while application performance data can show application score, users, hop count, average latency, percentile latency, baseline, and deviation. These views are useful when the scenario asks which ISP, application, or path should be examined first.

A disciplined anomaly workflow moves from broad to narrow. Start with the affected time range, geography, and application. Confirm whether users are actually impacted. Inspect Last Mile ISP or ASN behavior. Then determine whether the next action belongs to service desk triage, network operations, application owners, or provider escalation. The sequence matters because it prevents a graph from becoming a guessing engine.

> **🎓 Exam Tip**
> *Anomaly impact*
>
> Prioritize anomalies that overlap with poor ZDX Scores, affected users, and the reported time window. A red or elevated network path with no user impact may be worth watching, but it is not automatically the incident to chase first.

> **⚠️ Warning**
> *Baseline tunnel vision*
>
> Baseline deviation is powerful, but it is not the whole story. Always pair baseline deviation with user impact, application scope, and the number of affected users before deciding severity or escalation.

Differentiating Network vs Application Issues

Differentiating network and application issues begins with the selected application, the affected users, and the time window. A score drop for one application across multiple unrelated geographies may suggest application-side or hosting-side behavior. A score drop that follows one ISP, city, or path segment is more consistent with network investigation.

Web probe metrics help identify application-facing symptoms. Page Fetch Time measures the time to fetch the top-level page document. DNS Resolve Time identifies name-resolution delay. Server Response Time reflects time to first byte. Availability indicates whether the expected response was received. These metrics help separate slow application response from network path degradation.

Application incident metrics provide additional evidence when the application itself is suspect. HTTP errors, the ratio between server response time and page fetch time, last server leg latency, and redirect count can show whether users are encountering application errors, slow server response, excessive redirect behavior, or latency closer to the application side.

Cloud Path evidence is the counterbalance to application telemetry. If the application score is poor but Cloud Path shows elevated latency or loss on a specific segment, the issue may be network-side even though the user reports it as “the app is slow.” Users describe symptoms; ZDX helps identify layers.

Comparing multiple applications is one of the cleanest ways to separate network from application impact. If several unrelated applications degrade for the same population at the same time, a shared access path, ISP, or Service Edge route becomes more likely. If only one application degrades for the same users while other applications remain healthy, application-specific investigation becomes more appropriate.

User Details should be used when the scope narrows to an individual or small set of users. Device health metrics, CPU usage, memory usage, disk behavior, Wi-Fi signal quality, network adapter changes, and user device events can explain poor experience without blaming the ISP or the application. This is where many escalations should quietly retire.

The Incidents Dashboard helps when multiple users are affected. Its incident categories, including Device, Wi-Fi, Last Mile ISP, Intermediate ISP, ZIA Public Service Edge, ZPA, and Application, provide a starting classification for multi-user impact. The category should guide the next investigation step, not replace validation.

For ZIA and ZPA paths, use the Service Edge and Private Access context to understand where the measured path changes. ZIA Public Service Edge evidence can include impacted score, hop latency around the Service Edge, transaction behavior, connectivity errors, and probe errors. ZPA-related evidence can include App Connector or ZPA Public Service Edge path indicators. Keep the conclusion tied to metrics, not product assumptions.

When the current state is unclear, compare the degraded point against a known-good period. The comparison view can show differences across web, device, and Cloud Path metrics, which is often faster than manually scanning every graph. This is especially useful when the reported issue is intermittent or when the environment changed between a good and bad experience window.

Containment should follow classification. Device evidence leads to endpoint remediation. Wi-Fi evidence leads to local network action. Last Mile ISP evidence leads to provider-aware guidance or escalation. Intermediate ISP evidence leads to ASN/path escalation. Application evidence leads to the application owner with web probe and error metrics. Correct containment is not glamorous, but neither is sending three teams to fix one problem in the wrong place.

🔄 Cross-policy resources

ZIA, ZPA, Client Connector, and operations

ZDX is the experience evidence layer. ZIA, ZPA, Client Connector, identity, and ITSM processes may explain the access path, policy boundary, user population, or escalation workflow. Keep ZDX central, then bring in the adjacent system only when it materially changes the decision.

⚠️ Warning

Misclassification cost

The wrong classification wastes time twice: first by sending the issue to the wrong team, and again when the right team has to rebuild the evidence trail. Use score, scope, path, and metric family before escalating.

📝 Quick Review

- Start with scope: one user, one device, one location, one ISP, one ASN path, one application, or a widespread condition.
- Treat ZDX Score as a triage signal, not a standalone conclusion.
- Use the Performance Dashboard to identify affected application and geography before drilling into users.
- Use Network Intelligence when the scenario involves Last Mile ISP behavior, Intermediate ISP behavior, path anomalies, or baseline deviation.
- Separate Last Mile blackout evidence from Last Mile brownout evidence.
- Use ASN views to distinguish internal provider behavior from peering or inter-provider behavior.
- Pair anomaly data with user impact; not every baseline deviation is a user-impacting incident.
- Use Page Fetch Time, DNS Resolve Time, Server Response Time, and Availability to identify application-facing symptoms.
- Use Cloud Path evidence to locate network path segments that explain application slowness.
- Compare multiple applications to determine whether the problem follows an application or follows a shared network path.
- Use User Details and device health before escalating single-user problems externally.
- Avoid these common mistakes:
- Blaming the ISP just because the user is remote.
- Treating a highlighted network anomaly as an outage without checking poor scores and affected users.

- Trusting averages when P95 latency shows tail impact.
- Escalating to application owners without checking web probe and path evidence.
- Escalating to network teams without identifying the affected ISP, ASN, geography, time range, and user population.

🏅 Exam Blueprint Alignment

Scenario Review

This section maps most directly to the following scenario-based ZDXA objectives:

- Interpret the Performance Dashboard and scores to determine whether an issue is single-user, site-specific, regional, ISP-related, application-specific, or widespread.
- Use Application view to compare application performance and determine whether degradation follows one application or multiple applications.
- Analyze Last Mile ISP performance and distinguish connectivity-oriented blackout patterns from performance-oriented brownout patterns.
- Analyze Intermediate ISP performance, including Intra ASN, Inter ASN, and peering-related behavior.
- Identify network anomalies by correlating baseline deviation, latency, packet loss, probe count, affected users, and poor ZDX Scores.
- Analyze network path information to find bottlenecks across the end-to-end path.
- Differentiate device, network, Service Edge path, and application issues using ZDX telemetry rather than unsupported assumptions.

ZDX API Fundamentals

ZDX API fundamentals sit inside Monitoring and Triaging Digital Experience because the API is a force multiplier for triage. It gives programmatic access to ZDX features such as reporting, configuration, troubleshooting, and administration, but the purpose is still operational: get the right evidence to the right team quickly, with enough context to reduce MTTR and avoid unnecessary escalation.

At the administrator level, API fluency means understanding what the API can retrieve or initiate, which credentials and permissions are required, and what guardrails keep an integration safe. A candidate does not need to memorize every endpoint path to reason through a scenario. The important skill is knowing whether a use case belongs in the portal, an alert, a webhook, a scheduled reporting workflow, or a controlled troubleshooting automation.

The API is also part of the governance model. Before an external system pulls ZDX data or starts a troubleshooting action, an administrator should ensure that API access is backed by a role with only the required permissions. A reporting integration should not inherit the same privileges as an operations workflow that can start Deep Tracing. Least privilege is not paperwork; it is blast-radius control.

When the blueprint references ZDX API significance, read that as an expectation that candidates can explain why programmatic access matters. The API enables integrations with business intelligence tools, ITSM platforms, SIEM workflows, developer tools, and automation services. The exam scenario will usually be about choosing the right operational pattern, not writing code line by line.

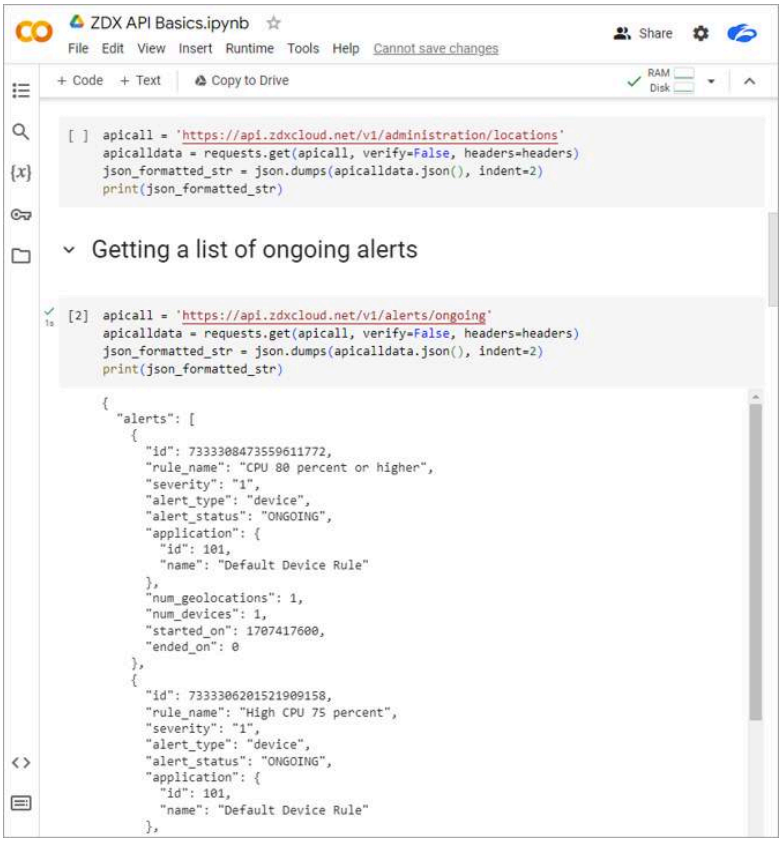
API Overview and Capabilities

The ZDX API provides programmatic access to ZDX data and selected ZDX actions. It is best understood as an extension of the operating model exposed through the Admin Portal: data about applications, users, devices, metrics, and troubleshooting can be consumed by external systems, while certain workflows can be initiated without requiring an administrator to manually remain in the portal.

The endpoint families map to the major operational needs of a ZDX deployment. Authentication returns a token for API access. Reporting retrieves ZDX Scores, application information, user information, device health metrics, and device events. Troubleshooting can initiate deeper diagnostic workflows such as Deep Tracing for a specific user and device. Administration and configuration functions support access to tenant objects such as locations, departments, users, roles, applications, and probes.

Authentication follows a token-based model. An administrator generates an API Key ID and API Key Secret for the client application. The external application exchanges those values at the token endpoint and receives a JSON Web Token. Subsequent API calls present that token, and optional callbacks can be used when an integration needs asynchronous behavior.

The key design point is that an API key should represent an application or integration, not a person casually reusing their own admin credentials. Treat the API Key Secret like any other production secret: store it securely, rotate it when required, avoid embedding it in scripts or notebooks, and separate keys by integration so that one compromised workflow does not expose every downstream system.



Reporting endpoints are the most common starting point because they support recurring operational questions. An external system can retrieve ZDX Scores and supporting data for applications, users, and devices so that a dashboard or report can show the current or historical digital experience. This is useful for operations reviews, service health summaries, and incident enrichment.

API reporting should be designed around the same dimensions used in ZDX analysis: application, user, device, department, location, geolocation, and time range. A report that pulls a score without the scope that produced it is easy to misread. A score tied to the wrong time window is worse than no score at all because it can send the escalation train to the wrong station.

There are two important data caveats. First, API results can differ slightly from UI values because aggregated metrics can be computed with approximate functions. Second, Client Connector telemetry can introduce a delay between collection and reporting. A well-designed integration accounts for that delay instead of treating every response as an instantaneous state of the world.

🎓 Exam Note

API timing and precision

- Do not overfit an automation rule to a single API sample. For scheduled reporting, align timestamps, account for collection-to-reporting delay, and expect minor differences between API-derived aggregates and dashboard values. For active incidents, use the API to enrich and correlate evidence, not to replace operator judgment.
- Configuration endpoints can support ZDX applications and probes, which makes them powerful and potentially risky. Changing what is monitored changes the data

that downstream teams trust. For administrator-level scenarios, the safest answer is usually to use configuration automation only when there is a clear naming standard, approval process, rollback path, and validation method.

- Troubleshooting endpoints add a different class of capability. Starting Deep Tracing or initiating score analysis through an API can be valuable when a ticket has enough evidence to justify deeper collection. This is especially useful when a service desk workflow needs to gather focused data for one user, one device, and one application without asking the user to reproduce the issue across three screens and a minor lunar phase.
- A Deep Tracing request should be targeted. The administrator should know the affected user, device, application, and duration, and should understand whether web probe, Cloud Path, and device statistics are needed. The goal is to collect enough detail to isolate the issue without turning diagnostics into background noise or unnecessary endpoint load.
- Machine-learning-assisted root cause analysis can also fit into an automated workflow when the evidence points to a poor ZDX Score and the team needs a probable cause. The important operational habit is to verify the analysis against supporting metrics such as Page Fetch Time, DNS Resolve Time, Server Response Time, Cloud Path latency, packet loss, and device health. Automation can accelerate analysis; it should not turn “probable” into “proven” by executive decree.
- API calls are rate limited, and rate limits scale by subscription tier. The design implication is straightforward: build integrations that batch, cache, back off, and respect rate-limit headers. Avoid polling every object at aggressive intervals just because a script can do it. ZDX is an operational telemetry system, not a free espresso machine for infinite queries.
- The ZDX API is not available on every subscription level, so an implementation plan should confirm licensing before promising an integration to the help desk, operations team, or reporting team. This is also a useful exam discriminator: when a scenario asks for an API-driven outcome, verify that the plan supports API access before selecting a workflow that depends on it.
- The API should be viewed as complementary to dashboards. The Performance Dashboard and Applications Overview are still the best places to visually establish scope and trend. The API becomes valuable when that evidence must be exported, correlated, enriched, ticketed, trended, or acted on by another system.
- Administrator access matters because the API can expose user, device, and operational data. Use roles that match the workflow, and use obfuscation where teams do not need personally identifiable user or device details. A global operations report may need location and application trends; it may not need the name of every affected employee.

⚠️ Warning

Automation without scope

A common failure pattern is building an integration that treats every low score as the same incident. A low ZDX Score for one remote user, a poor score for all users in one city, and a global application degradation require different actions. Automations should preserve scope, not flatten it.

Use Cases for Automation and Reporting

The simplest API use case is scheduled reporting. A business intelligence platform can consume ZDX data to produce recurring summaries of application experience, regional trends, device health, and operational performance. The best reports do not merely repeat the dashboard; they help teams see whether user experience is improving, degrading, or shifting by geography, application, or user population.

For reporting, select a refresh cadence that matches how ZDX data is collected and processed. Pulling too frequently can waste rate limit and create false urgency. Pulling too slowly can miss operational changes that matter. A practical design separates near-real-time incident enrichment from daily or weekly trend reporting.

In a scenario where a single SaaS application is slow in one country, the first move is still to establish scope through the performance view and application-specific context. Once the affected geography is confirmed, the API can help retrieve impacted users, devices, and supporting metrics for the same application and time range. That evidence can then populate a ticket or incident record automatically.

Device-health reporting is another strong use case. ZDX collects endpoint metrics such as CPU usage, memory usage, disk activity, network I/O, Wi-Fi signal strength, and device events. A reporting workflow can help desktop teams identify patterns such as repeated endpoint saturation, weak Wi-Fi environments, or recurring device events that align with poor experience.

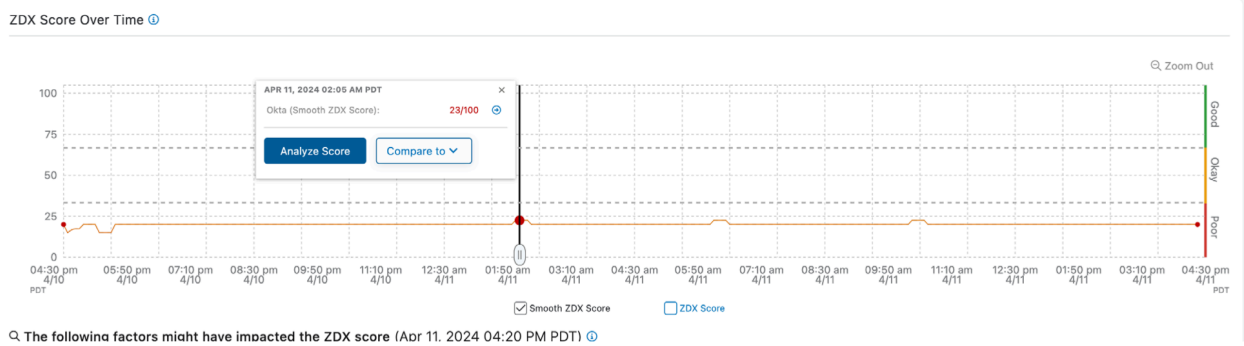
The same logic applies to application reporting. Page Fetch Time, DNS Resolve Time, Server Response Time, and availability help separate application-side degradation from local network or endpoint problems. When these values are pulled into a reporting system, preserve the metric names and units so that downstream teams do not accidentally compare milliseconds, percentages, and scores as if they were interchangeable. That is how dashboards become confetti cannons.

Troubleshooting automation is more sensitive than reporting automation. Starting a Deep Tracing session through the API can shorten the time between ticket intake and evidence collection, but it should be gated by criteria. For example, use it when the ticket identifies the user, device, and application, and when dashboard or API evidence suggests that standard telemetry is not enough.

An API-triggered Deep Tracing workflow should also define the duration and data needed. Short sessions are often enough to capture web probe, Cloud Path, and device statistics for a reproducible issue. Longer sessions should be reserved for persistent or intermittent problems where the additional collection time is justified.

Automated root cause analysis is valuable when the workflow needs a probable cause quickly. For example, if a user’s poor score correlates with high latency on a specific Cloud Path segment or heavy CPU usage on the endpoint, the analysis can help the service desk route the

ticket to the correct resolver group. The routing decision should still be checked against the supporting telemetry.



Alert enrichment is another practical use case. An alert can notify the operations team that a threshold or dynamic condition has been violated, while the API can retrieve additional details for the affected application, user group, devices, or region. This combination is stronger than either approach alone: the alert provides immediacy, and the API provides context.

For executive or operational review, APIs can feed broader reporting workflows that summarize ZDX Scores by application, location, and user population. The key is to report trends and exceptions, not every raw metric. Leaders need to know where experience is degrading, which teams are affected, and whether the issue is tied to applications, networks, or devices.

Containment in ZDX is operational containment. The objective is to limit confusion and unnecessary work by identifying exactly who is affected and what evidence supports the scope. An API-backed workflow can help notify only the impacted region, route a ticket to the right team, and avoid asking unaffected users to take pointless remediation steps.

Automation should never erase the human review path. Build workflows that preserve links back to the ZDX evidence, include the time range evaluated, identify the application and device context, and record what action was taken. That audit trail becomes essential when a ticket escalates or when teams need to tune the workflow after a noisy incident.

🎓 Exam Tip

Choosing portal, pull, or push

Use the portal when an analyst needs visual triage and exploratory investigation. Use API pull workflows when another system needs recurring ZDX data or ticket enrichment. Use webhook push workflows when an alert should create or update an external incident in near real time. The scenario usually tells you which pattern fits if you pay attention to timing, scope, and ownership.

Integrating ZDX Data into External Systems

Integrating ZDX data into external systems is about operational continuity. If the service desk lives in an ITSM platform, the incident context should arrive there. If operations reviews are driven from a business intelligence tool, ZDX data should be represented there with the same scope and timing used in the portal. If a security or operations platform correlates events, ZDX telemetry can add user-experience context.

There are two common integration patterns: event-driven and query-driven. Event-driven integrations use alerts and webhooks to notify an external system when a rule is triggered. Query-driven integrations use the API to retrieve data on demand or on a schedule. A mature workflow often uses both: webhooks create or update the incident, and API calls enrich it with user, device, application, and metric details.

ServiceNow-style ITSM integration is the easiest pattern to reason about. ZDX alerts and metrics can be made available in the ITSM console so that service desk analysts can see impacted users, locations, and supporting telemetry without switching tools. In advanced workflows, analysts can also initiate diagnostic actions such as Deep Tracing or root cause analysis from the ITSM process.

A webhook payload should be treated as the incident trigger, not the entire investigation. Alert details can include information such as rule name, severity, status, affected users, affected locations, and a pointer to alert details. The receiving system should then decide whether to open a new ticket, update an existing one, suppress a duplicate, or escalate based on operational policy.

For SIEM or operations analytics systems, ZDX data should be mapped carefully. ZDX telemetry is digital experience evidence; it may help explain user impact, service degradation, or network anomalies, but it should not automatically be treated as a security incident. Severity mapping should reflect operational impact and confidence, not just the presence of a low score.

Business intelligence integrations should normalize time windows and dimensions. A dashboard that compares application performance across regions should use consistent reporting intervals and should distinguish between application, location, department, and user-level scores. Mixing these levels without labels creates misleading trends.

Authentication and authorization must be designed separately for APIs and webhooks. API access uses the API Key ID and Secret to obtain a token for calls into ZDX. Webhook delivery to an external receiver may use its own authentication method, such as simple authentication or bearer token authentication, depending on the receiving platform. Do not reuse secrets across these two directions just because both say “integration.”

Least privilege applies to every integration persona. A reporting platform may only need read access to reports. An ITSM integration that starts diagnostics needs a narrower but more sensitive privilege set. An administrator configuring applications and probes requires different access than a service desk analyst reviewing user experience. Align the role to the task.

🔄 Cross-policy resources

ZDX with ZIA, ZPA, and Client Connector

- ZDX data is most useful when it is interpreted with the forwarding path. Zscaler Client Connector supplies the endpoint telemetry and probes. ZIA and ZPA paths can introduce Service Edge and App Connector context in Cloud Path views. API integrations should preserve that context so tickets route to the correct desktop, network, application, or security team.
- Cloud Path context is especially important for integrated workflows. If an issue appears between the user and local egress, a network or desktop queue may be appropriate. If the issue appears between egress and a ZIA or ZPA Service Edge, network path and service reachability become the focus. If the issue appears beyond the Service Edge toward the application, the application owner or hosting provider may need to investigate.
- Call quality monitoring integrations add another layer. For UCaaS scenarios, meeting quality, MOS, latency, jitter, packet loss, Cloud Path, and device metrics can be combined to explain why a meeting felt poor. An external ticket should keep those signals together so that a voice/video issue is not misrouted as a generic web application ticket.
- Validation is not optional. After integrating ZDX data into an external system, compare the external record with the ZDX Alerts page, Performance Dashboard, Applications Overview, or User Details view for the same time range. Small aggregate differences can be expected, but the scope, severity, affected application, and impacted population should tell the same operational story.
- Good integrations also handle failure gracefully. They respect rate-limit headers, retry with backoff, avoid duplicate ticket storms, and log failed requests without exposing secrets. When a webhook receiver is unavailable, the workflow should fail in a way that can be detected and corrected, not silently drop the only signal anyone was counting on.
- The final design criterion is ownership. An API or webhook integration should make it clearer who acts next. If the workflow creates tickets that no team owns, dumps raw metrics into a queue, or starts diagnostics with no analyst assigned, it has automated confusion. The best integrations turn ZDX evidence into accountable action.
- [Note: Integration design discipline]
  A useful ZDX integration answers four questions every time: What changed, who is affected, where is the evidence, and who owns the next action? If the workflow cannot answer those questions, simplify it before expanding it.

📝 Quick Review

- The ZDX API provides programmatic access to ZDX capabilities for reporting, configuration, troubleshooting, and administration.
- API authentication uses an API Key ID and API Key Secret to obtain a token that is used for subsequent calls.
- Reporting workflows should preserve time range, application, user, device, department, location, and geolocation context.
- API-derived data can differ slightly from UI aggregates and can be delayed because

telemetry is collected and processed before reporting.

- Use webhooks for event-driven alert delivery and API calls for enrichment, reporting, or controlled troubleshooting actions.
- Deep Tracing and root cause analysis can be initiated through automation only when the workflow is scoped to a specific user, device, application, and operational need.
- Rate limits and subscription availability must be considered before designing polling-heavy integrations.
- Least privilege is mandatory: reporting integrations, ITSM workflows, and diagnostic automations should not share the same permissions.
- Preserve ZIA, ZPA, Service Edge, Client Connector, and application context when routing incidents externally.
- Common mistakes to avoid: polling too aggressively, treating one low score as a widespread incident, ignoring telemetry delay, exposing API secrets in scripts, over-permissioning service accounts, starting diagnostics broadly, and creating tickets without clear ownership.

🏅 Exam Blueprint Alignment

Scenario Review

- Section 1.9: Understand the significance of the ZDX API for programmatic access to ZDX features.
- Section 1.7: Understand the value and benefits of common third-party integrations, including call quality monitoring, ServiceNow-style ITSM workflows, and API-based data access.
- Section 1.2: Interpret dashboard scope and scores, then use API or integration data to identify impacted users, locations, applications, and escalation paths.
- Section 3.2: Analyze diagnostics session output when Deep Tracing or related diagnostic workflows are initiated through an approved operational process.
- Section 3.4: Use Zscaler AI-assisted analysis to accelerate root cause investigation, while validating the probable cause against ZDX metrics.
- Section 3.5: Use predefined, customized, or API-fed reports to analyze organization-level performance trends.
- Section 4.1: Apply role-based access control so service desk, operations, reporting, and diagnostic workflows receive only the permissions they require.
- Section 4.2: Configure alert rules and notification types, and understand how token-based API authentication and webhooks support external ITSM or incident-management integrations.

Operational Configuration and Optimization (30%)

Configuring Applications and Monitoring Models

This subsection maps most closely to the Operational Configuration and Optimization objectives that ask candidates to configure predefined applications, custom applications, Hosted Monitoring, and Real User Monitoring based on a scenario. It also supports probe-selection objectives because the monitoring model is incomplete until the candidate can explain why a Web Probe, Cloud Path Probe, Hosted Probe, or RUM data source is appropriate.

An application is the monitoring boundary. Collections organize related applications and probes. Probes provide the measurements. Web Probes measure application responsiveness and availability. Cloud Path Probes measure path behavior between the device or hosted location and the destination. Hosted Probes provide a Zscaler-cloud vantage point. RUM provides browser-session evidence from actual user interactions. The administrator must choose the model before tuning the knobs.

Before configuring, define four things: the service boundary, the users who actually consume the service, the vantage point that matters, and the metric that would prove success or failure. A sales SaaS application used worldwide, an internal HR portal used only through ZPA, and a browser-heavy claims application may all be “applications,” but they should not be monitored as if they fail in the same way.

After configuring, validate before declaring victory. Activate the changes, confirm probe status, allow enough time for telemetry to appear, and review the relevant dashboard for the expected users, locations, departments, and devices. If a dashboard looks empty or misleading, first check scope, subscription support, role visibility, endpoint readiness, and probe success criteria. Do not start with “the platform is broken.” That is a troubleshooting tax nobody wants to pay.

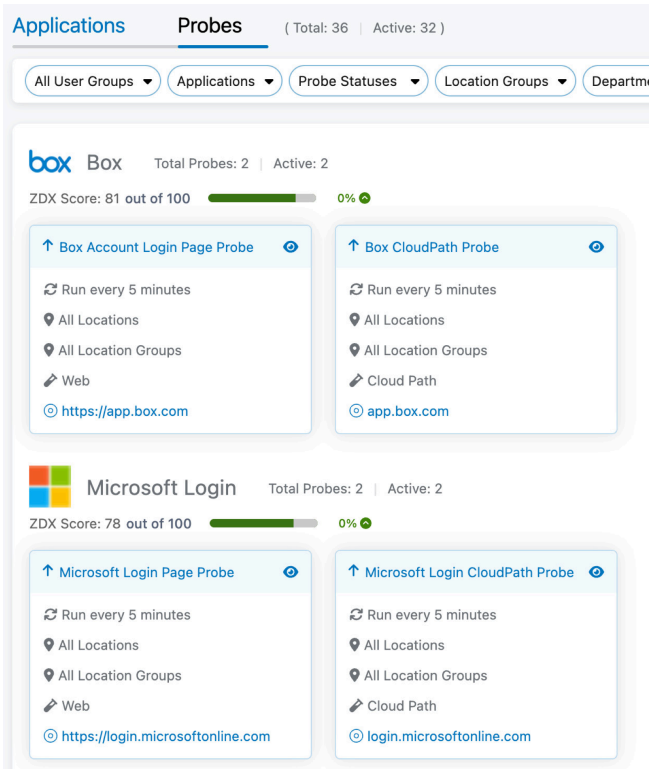
🎓 Exam Note

Application objects and evidence quality

A ZDX application should represent a service boundary that operations can act on. If the object mixes unrelated services, regions, or ownership models, the resulting score can be technically valid but operationally confusing.

Predefined Applications

Predefined applications are the preferred starting point when the monitored service is already modeled in ZDX. They reduce setup effort for common enterprise SaaS and collaboration platforms because the application framework and expected probe patterns are already available. In a scenario, a supported SaaS application with standard monitoring requirements should usually point the administrator toward predefined application onboarding first.

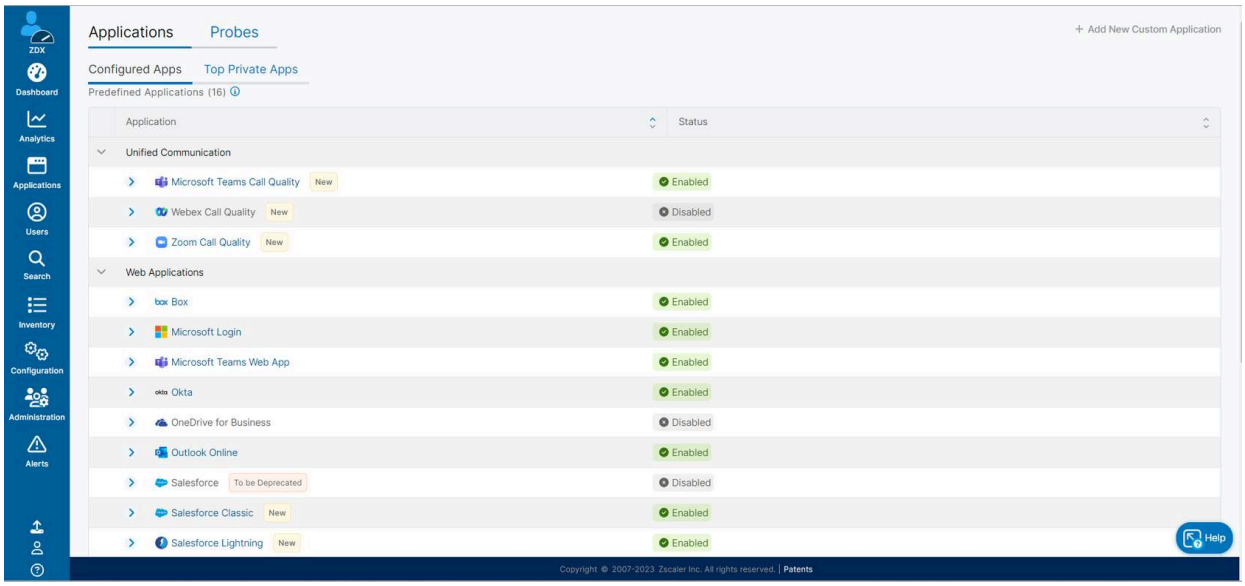


The main design check is fit. The available predefined applications depend on subscription level, and some applications require tenant-specific information before the telemetry represents the organization's real user path. A candidate should not assume that enabling a predefined object automatically validates tenant alignment, entitlement, or user impact. It only creates the starting point for monitoring.

When onboarded, predefined applications commonly include preconfigured Web and Cloud Path probes. That combination gives the operations team both application responsiveness and network-path visibility. The administrator should confirm that expected probe data appears and that impacted locations, regions, departments, and users make sense for the selected application.

Subscription scope matters in exam scenarios. A Microsoft 365-focused deployment may expose a different predefined application set than a broader Advanced deployment. UCaaS capabilities may also vary by entitlement. When a scenario includes a requirement such as Zoom Call Quality, Microsoft Teams Call Quality, or Webex monitoring, verify that the application and monitoring type are supported before designing the operational workflow around it.

Unified communications applications require more precise thinking than ordinary web applications. Monitoring the Teams web application front door is not the same as analyzing meeting quality. Call-quality integrations bring meeting or media-related metrics into the ZDX workflow, while the web application probe focuses on application access and responsiveness. The administrator should match the monitoring model to the user complaint: "Teams will not load" and "my meeting audio is bad" are cousins, not twins.



Additional probes can be added where supported, but they should serve a specific purpose. A tenant-specific front door, a regional application endpoint, or a business-critical workflow may justify additional monitoring. Adding probes without a reason can create noise, consume limits, and make troubleshooting harder.

> ⚠️ **Warning**
> *Predefined does not mean maintenance-free*
>
> Predefined applications accelerate onboarding, but the administrator still owns validation. Check tenant information, probe status, subscription support, scope, and dashboard evidence before assuming the configuration is complete.

Custom Applications

Custom applications are used when the service is not adequately represented by a predefined application. That includes internal web applications, externally hosted business applications, private applications accessed through ZPA, and SaaS services that require organization-specific monitoring behavior. Custom applications let the administrator model what the business actually depends on, rather than forcing a unique service into a generic shape.

For most custom web applications, at least one Web Probe is required before meaningful availability and performance data can be collected. Network applications are the important exception: they do not require a web front end and can be monitored with a single Cloud Path Probe. This distinction matters in scenarios where the application is not HTTP-based or where the requirement is path monitoring rather than page monitoring.

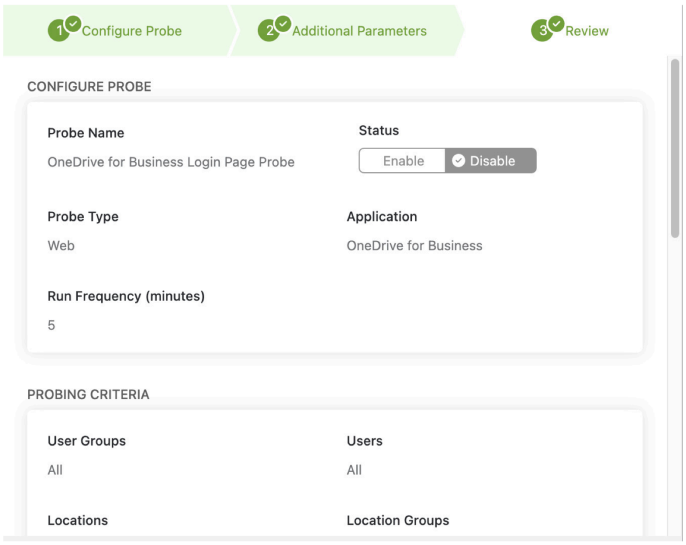
The application object should have a clear name, useful description, and intentional status. If the application is disabled, its probes are disabled as well. A practical naming standard should

make the application understandable to the service desk, network team, and application owner, not only to the admin who created it during a “quick fix” that somehow became permanent.

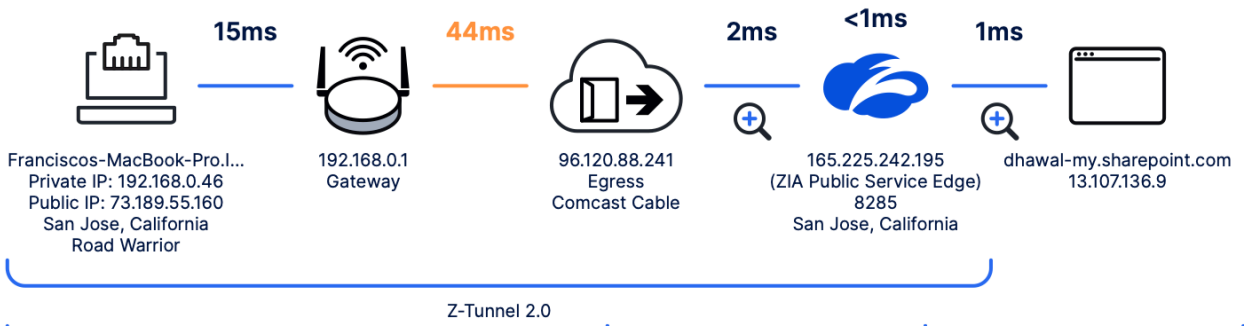
A Web Probe should target the URL that best represents application availability from the user’s perspective. The resulting metrics help separate DNS resolution time, server response time, Page Fetch Time, and availability behavior. If the configured URL does not represent the actual user entry point, the probe can still run successfully while measuring the wrong operational condition.

HTTP response behavior must be configured with care. Many enterprise applications redirect unauthenticated users to an identity provider, so a redirect may be a valid success condition. The default success set includes informational, success, and redirection responses, but administrators should tune accepted codes only when the application behavior justifies it. A probe that treats too many responses as success can hide a real failure; a probe that is too strict can create false alarms.

Advanced Web Probe settings are for known application behaviors, not guesswork. Request headers can support applications that expect specific values. Attempts and timeout can be adjusted for applications that are expected to respond slowly.
Follow-redirect behavior and maximum redirects can be tuned when redirect chains are part of the normal experience. Each adjustment should make the probe more truthful, not merely less noisy.

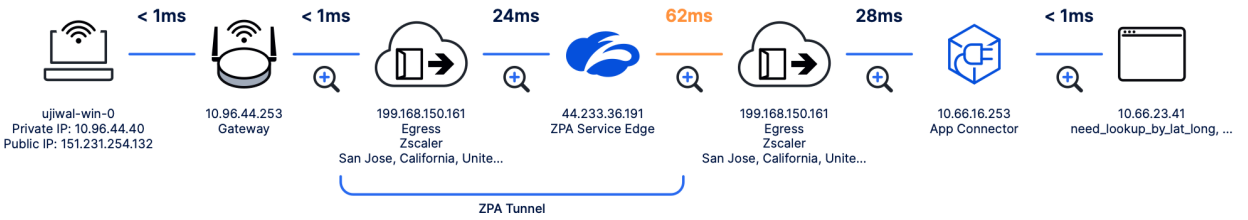


Cloud Path Probes add network-path evidence by measuring hop count, packet loss, and latency along the route to the destination. They can use ICMP, TCP, UDP, or Adaptive protocol behavior depending on design requirements. A Cloud Path Probe is especially useful when Page Fetch Time is elevated and the administrator needs to determine whether the path, rather than the application front door, is contributing to the degradation.



Probe scoping is one of the highest-value optimization decisions for custom applications. Use include and exclude criteria so that only relevant users, groups, departments, locations, or devices run probes. This is especially important for private applications and narrowly used internal services. Monitoring every user for an application only ten people use is not diligence; it is telemetry with poor manners.

Top private application visibility can help identify ZPA-hosted services that deserve ZDX monitoring. When private application usage is visible by port, protocol, user count, or bandwidth, administrators can prioritize high-value candidates and configure probes for the services that matter most. If an application is already configured for ZDX, manage it through the existing probe configuration instead of duplicating the monitoring object.



Application modeling should avoid combining unrelated services into one custom application. If a service has separate regional instances, different ownership, or different escalation paths, consider whether separate ZDX applications would produce cleaner evidence. A single application object that blends US, EMEA, and APAC instances can turn a regional issue into a vague global average. That is dashboard fog, not observability.

🎓 Exam Tip

Web Probe versus Cloud Path Probe

Use a Web Probe when the scenario asks whether the application responds and how long the web transaction takes. Use a Cloud Path Probe when the scenario asks where latency, loss, or hop-related degradation occurs. For “the app is slow” scenarios, expect to correlate both.

Managed Monitoring and Hosted Monitoring

Hosted Monitoring is the managed monitoring model used when the required vantage point is the Zscaler cloud rather than an end-user device. It operates from Zscaler cloud infrastructure as a multi-tenant service and uses Zscaler Hosted Probes to monitor a specific service or application directly from a Zscaler data center to an endpoint destination. In the blueprint, this belongs beside predefined applications, custom applications, and RUM because it is another way to decide how ZDX should observe application performance.

The key design distinction is perspective. Endpoint-based ZDX probes show what users experience from their devices, including local network, Wi-Fi, ISP, forwarding path, and endpoint conditions. Hosted Monitoring shows what selected Zscaler hosted locations observe when testing the destination. If endpoint users are degraded but hosted probes remain healthy, the issue is more likely closer to the user side of the path. If both endpoint and hosted views degrade, the application, destination network, or shared upstream path becomes a stronger candidate.

Hosted Monitoring uses independent collections to organize Web and Cloud Path probes. A collection should be designed around an operational purpose, such as a service family, business-critical application, regional service, or application owner. Administrators should also select hosted locations intentionally. Although all hosted locations may be selected by default, a scenario may call for testing only the regions that matter to the users, application, or incident pattern being investigated.

For Hosted Web Probes, ZDX displays Availability and Page Fetch Time by default. Availability can be reviewed for Month To Date, Past 30 Days, or the selected time range, with comparison against the prior period. Page Fetch Time can be drilled into by hosted location, and detailed run metrics include Redirect Time, DNS, TCP, SSL Handshake, Server Response Time, and Page Fetch Time. This makes Hosted Web Probes useful when the question is whether the destination responds from the Zscaler cloud and which part of the web transaction contributes to delay.

For Hosted Cloud Path Probes, ZDX displays End-to-End Latency by default. Administrators can add Cloud Path metrics such as packet loss and jitter, inspect individual probe runs, and review maximum hop count. Cloud Path Visualization consolidates hosted locations into a path representation so the administrator can inspect route details by path leg. This is the better hosted model when the scenario asks about network-path behavior rather than only application responsiveness.

Time range selection changes how evidence should be interpreted. Hosted Monitoring provides a longer default view for the initial chart and lets administrators focus the widget into shorter operational windows. The time range widget supports focused analysis from 2 to 48 hours, with 24 hours as the default widget range. Scatter view is especially useful for intermittent issues because each data point represents an individual probe run rather than only a trend line.

🎓 Exam Tip

Hosted Monitoring decision logic

- Choose Hosted Monitoring when the scenario asks for a Zscaler-cloud vantage point, continuous service validation, or comparison between endpoint-observed degradation and cloud-originated tests. Do not use hosted results alone to conclude that users are unaffected; Hosted Monitoring does not include endpoint health, browser-session behavior, Wi-Fi, local ISP, or device event context.
- Hosted Monitoring is most valuable when paired with the rest of the ZDX evidence chain. Use it to validate service reachability and cloud-side path behavior, then compare those findings with endpoint probes, Application view, User Details, Cloud Path, RUM, and device telemetry as the scenario requires. The administrator's job is to avoid overclaiming from one viewpoint. One vantage point gives evidence; correlated vantage points give confidence.

Real User Monitoring (RUM)

Real User Monitoring captures browser-based interactions with applications on supported Windows and macOS devices. Unlike synthetic probes, which test a configured transaction at a defined interval, RUM observes actual browser sessions. It is most valuable when the application is reachable but the user's lived experience is still poor.

Importantly, only sessions for configured applications are recorded.

RUM provides page-level performance metrics such as Page Load (the time it takes for the page to actually load) and DOM Load (the time for the page to be interactable) times, and Core Web Vitals, including Largest Contentful Paint. Those metrics help identify bottlenecks that occur during browser rendering, page composition, or client-side application behavior. A synthetic probe can confirm the front door responded; RUM helps explain whether the page experience was actually usable.

The prerequisites are part of the design. RUM requires the appropriate subscription level, role permissions that allow configuration and viewing, compatible Zscaler Client Connector and ZDX Module versions, a supported browser extension, and endpoint allowlisting for the required RUM process. When RUM data is missing, these dependencies should be checked before changing the application model.

At scale, RUM extension deployment should be handled through endpoint management tooling rather than manual user installation. Consistency matters because browser-session data is only available from devices that have the required components in place. RUM without disciplined endpoint readiness is like asking for packet captures from a laptop that is still in the box.

When enabling RUM for an application, the administrator defines the relevant domain and session behavior, then activates the change. For custom applications, RUM can be part of the same application design discussion as Web and Cloud Path probes. A custom RUM collection

can help operations staff organize and review browser-experience monitoring separately from other application models.

RUM data can appear in application, user, and performance-oriented views, depending on subscription and role permissions. That visibility model is important in scenarios where one administrator can configure the application but another cannot see the resulting dashboard data. Lack of visibility may be an RBAC design issue, not a telemetry failure.

RUM does not replace Web Probes or Cloud Path Probes. It complements them. Use RUM when the scenario focuses on actual browser experience, page loading, or Core Web Vitals. Use Web Probes for synthetic responsiveness and availability. Use Cloud Path Probes for network-path attribution. A real deployment often needs more than one of these views to tell the whole story.

🔄 Cross-policy resources

RUM dependencies

RUM depends on application configuration, Client Connector and ZDX Module compatibility, browser extension deployment, endpoint allowlisting, and ZDX role permissions. Treat those dependencies as part of the monitoring design, not as afterthoughts.

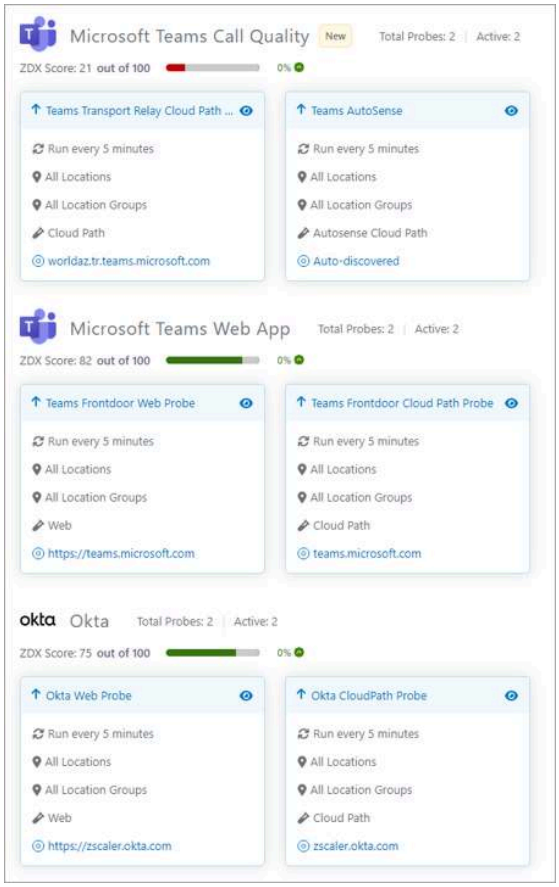
Selecting the Right Monitoring Approach

The fastest way to choose the right monitoring model is to identify the required vantage point. If the service is a supported SaaS application, start with a predefined application.

Note that RUM can be created standalone or with an existing SaaS application, but not with a network only application.

If the service is organization-specific, use a custom application. If the question is application responsiveness, use a Web Probe. If the question is path behavior, use a Cloud Path Probe. If the question is Zscaler-cloud visibility, use Hosted Monitoring. If the question is real browser experience, use RUM.

Predefined applications are strongest when the service is supported and the organization wants fast onboarding with known monitoring behavior. Custom applications are strongest when the service, URL, tenant behavior, or population is unique. A scenario that names a common



SaaS product does not automatically require a custom application unless the requirement includes a nonstandard workflow or tenant-specific path that the predefined model does not cover.

Web Probes and Cloud Path Probes answer different questions. Web Probe metrics help determine whether the application responded, how DNS behaved, when the server returned the first byte, and how long the page fetch took. Cloud Path metrics help locate latency and loss along the route. When Page Fetch Time rises while Cloud Path remains clean, the application or server side deserves attention. When path latency or destination loss rises, the network investigation gets warmer.

Hosted Monitoring runs probes from within the Zscaler cloud infrastructure to a destination service. This gives administrators a cloud-based vantage point that is independent of a specific user device. It is useful for continuous service validation, off-hours monitoring, and determining whether a service is reachable from selected Zscaler hosted locations.

Hosted Web Probes provide availability and Page Fetch Time data from the selected hosted locations. Hosted Cloud Path Probes provide end-to-end latency, packet loss, jitter, and hop count. This makes Hosted Monitoring useful when the operations team wants to compare endpoint-observed degradation with a Zscaler-cloud-originated measurement. Different vantage points can expose different fault domains.

Endpoint-based probes and Hosted Monitoring should not be treated as interchangeable. Endpoint probes show what users experience from their device and local network. Hosted probes show what the Zscaler cloud observes from selected data centers. If endpoint users in one region are degraded but hosted probes look normal, the issue may be closer to the users, their local networks, their ISP, or the access path. If both endpoint and hosted views degrade, the application or shared upstream path deserves stronger scrutiny.

RUM is the right choice when the complaint is about real browser use rather than synthetic reachability. A Web Probe may show that the top-level page responds, while RUM may show slow page load behavior or poor Core Web Vitals during actual sessions. In modern browser applications, that distinction can be the difference between “the app is up” and “the app is usable.” Those are not the same sentence, no matter how many dashboards try to make them rhyme.

Probe assignment should reflect business usage. Organization-wide critical applications may be assigned broadly. Department-specific applications should be scoped to the relevant department or group. Noisy or newly deployed applications may be monitored temporarily until the support pattern stabilizes. This keeps monitoring relevant while reducing unnecessary probe volume and dashboard clutter.

When validating a monitoring approach, follow a consistent sequence: confirm the application object, confirm probe status, confirm activation, wait for data, check scope, review the relevant dashboard, and inspect probe errors if the data does not match expectations. If the evidence

still does not align, compare user, application, Cloud Path, hosted, and RUM perspectives based on the scenario.

Operational ownership should be clear from the evidence. Web Probe and RUM findings often route toward application or endpoint teams. Cloud Path findings often route toward network, ISP, or Service Edge investigation. Device-health findings route toward endpoint operations. Hosted Monitoring can help separate user-side reachability from cloud-side or application-side conditions. The administrator's design should make that handoff easier.

For exam scenarios, avoid one-size-fits-all answers. "Enable monitoring" is not enough. The better answer identifies the application model, the probe type, the population scope, the validation view, and the reason that evidence answers the scenario. ZDXA candidates should be ready to explain why they chose the monitoring approach, not just where they clicked. The click path is temporary; the reasoning is what survives version changes.

🎓 Exam Tip

Scenario elimination logic

Supported SaaS with standard needs usually points to a predefined application. Internal or organization-specific web services point to custom applications. Path attribution points to Cloud Path. Zscaler-cloud vantage points point to Hosted Monitoring. Browser-session experience points to RUM.

📝 Quick Review

- Operational configuration is evidence design, not feature enablement.
- Use predefined applications when ZDX already models the SaaS or UCaaS service and the subscription supports it.
- Validate tenant alignment, subscription support, probe status, and dashboard data after enabling predefined applications.
- Use custom applications for internal, private, externally hosted, or organization-specific services.
- Most custom web applications require at least one Web Probe; network applications can be monitored with Cloud Path only.
- Web Probes measure application responsiveness and availability through metrics such as Page Fetch Time, DNS time, server response time, and HTTP response behavior.
- Cloud Path Probes measure path behavior such as latency, packet loss, hop count, and route segments.
- Hosted Monitoring runs probes from selected Zscaler hosted locations and provides a cloud-based vantage point.
- RUM captures actual browser-session performance, including page-loading and Core Web Vitals-style metrics.
- Scope probes to the users, groups, departments, locations, and devices that actually need them.
- Keep application objects aligned to actionable service boundaries; avoid mixing unrelated services or regions.

- Validate after activation and investigate scope, permissions, endpoint readiness, and probe success criteria before escalating.
- Common mistakes to avoid: wrong tenant information, disabled probes, unactivated changes, global scoping for narrow apps, overly broad HTTP success codes, treating intermediate hop loss as conclusive, and using synthetic probes alone for browser-rendering complaints.

🏅 Exam Blueprint Alignment

Scenario Review

- Section 2.1: Given a set of requirements, configure predefined applications, custom applications, Hosted Monitoring, and Real User Monitoring.
- Section 2.2: Define and differentiate Web Probes, Cloud Path Probes, and Hosted Probes; select the correct probe type for the scenario.
- Section 2.4: Given a set of requirements, monitor real-time application performance from a user perspective using browser-based performance metrics.
- Supporting overlap with Section 1.2: Interpret Performance Dashboard and application score data to determine whether impact is single-user, site, region, or widespread.
- Supporting overlap with Section 1.4: Use Application view to compare application performance and validate whether configured monitoring produces actionable evidence.

Configuring Probes

Configuring probes is the point where application inventory becomes operational telemetry. In ZDX, applications provide the monitoring container, while probes determine what is measured and from where. Predefined applications normally reduce configuration effort because standard probes are available when the application is onboarded. Custom applications require the engineer to define the monitoring target and the probe behavior intentionally, because ZDX cannot infer what “healthy” means for a custom service without configuration.

The blueprint scope for this subsection maps most directly to differentiating Web Probes, Cloud Path Probes, and Hosted Probes; configuring predefined and custom applications; and using monitoring from an end-user perspective. The scenario pattern is usually not “what button do you press?” It is closer to “given these requirements, which probe type and scope would produce the evidence needed to prove or disprove the suspected failure domain?”

Probe configuration has an object-model implication that matters in the field. A Web Probe validates the application transaction at the configured URL. A Cloud Path Probe characterizes the network path to a destination. A Hosted Probe changes the vantage point by running from Zscaler cloud infrastructure rather than from the user device. That difference in vantage point is the difference between proving what a user experiences and proving what a monitored service looks like from a Zscaler-hosted monitoring location.

A good implementation review asks four questions before any probe is considered production-ready: Is the target representative of the user experience? Is the probe scoped to the correct population? Are response codes, redirects, DNS behavior, and protocol settings aligned with the application design? Will the resulting telemetry help an operator decide whether the next action belongs with the endpoint team, the network team, the app owner, the ISP, or Zscaler Support?

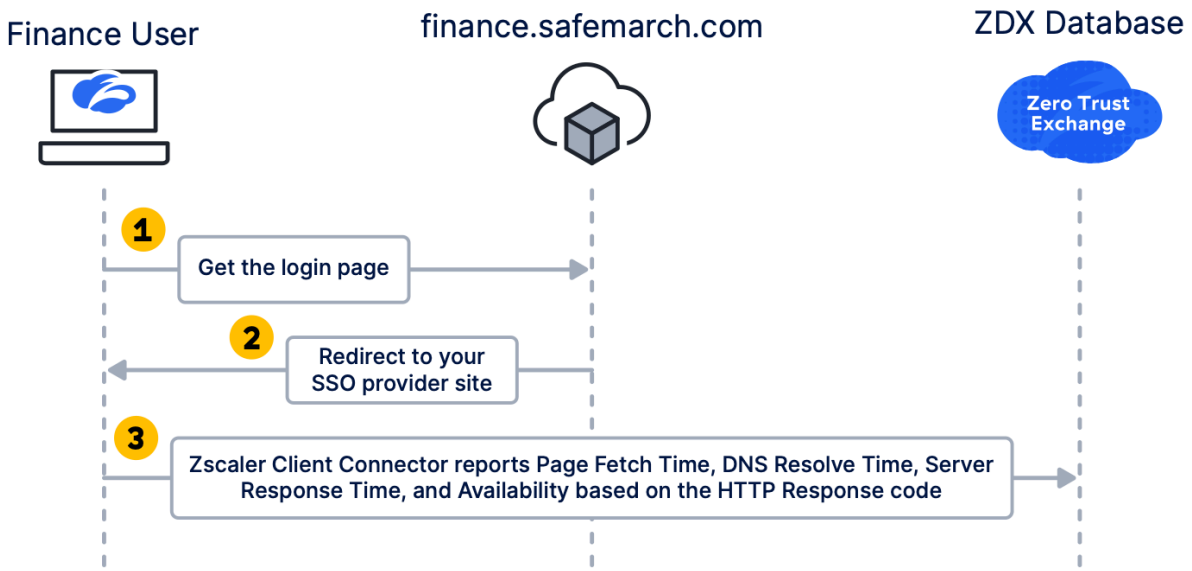
🎓 Exam Tip

Probe reasoning

Start with the operational question, not the probe name. If the question is “is the application responding correctly,” think Web Probe. If the question is “where is latency or loss introduced,” think Cloud Path Probe. If the question is “what does this service look like from a Zscaler cloud vantage point,” think Hosted Probe.

Web Probes

A Web Probe measures application availability and responsiveness by requesting the configured web destination. It is designed to represent a web transaction closely enough to provide actionable application telemetry, while staying bounded to the configured page rather than crawling every embedded object on the site. This distinction matters because candidates should not assume that a Web Probe measures the full browser-rendered experience of every image, script, and dependent service on a page.



Page Fetch Time is the central metric for Web Probe interpretation. It measures the network fetch time for the configured page, and it is one of the most important indicators when a user says an application feels slow. A low-latency network path does not rescue the experience if the page fetch takes too long, and a fast endpoint does not fix an application that is slow to respond. In exam scenarios, Page Fetch Time is often the first clue that the problem might be application-side, transport-side, or DNS-related depending on the companion metrics.

Web Probes also report DNS time, server response time, and availability. DNS time reflects how long name resolution takes for the configured host. Server response time, commonly understood as Time to First Byte, reflects the wait between the request and the first byte returned by the server. Availability is tied to the HTTP response behavior and whether the probe receives an expected response rather than timing out or failing before a usable response is returned.

Configuring a Web Probe requires the engineer to define what success means for the target application. For many web applications, a redirect to an identity provider is a healthy result, because the application front door is alive and correctly sending unauthenticated users to SSO. For another application, the same redirect could be misleading if the probe is supposed to

validate a specific tenant landing page. The correct response-code set is therefore not a trivia item; it is part of the monitoring contract.

Custom applications usually require more design attention than predefined applications because the engineer controls the probe name, target URL, response-code expectations, headers when required, redirects, timeout behavior, and scope. Request headers can be useful when the application expects a specific header value, but they should be treated carefully because a probe is not a substitute for full application authentication testing. The goal is to test a stable, representative endpoint that indicates whether the service is reachable and responsive.

Predefined applications reduce configuration burden but also limit what should be changed. Some default settings are intentionally fixed or constrained because the application pattern is already known. In a scenario where a predefined application probe cannot be modified the same way as a custom probe, the right conclusion is not that ZDX is broken. It is that preset probe behavior is part of the predefined application model, and optimization should focus on scope, enablement, and interpretation.

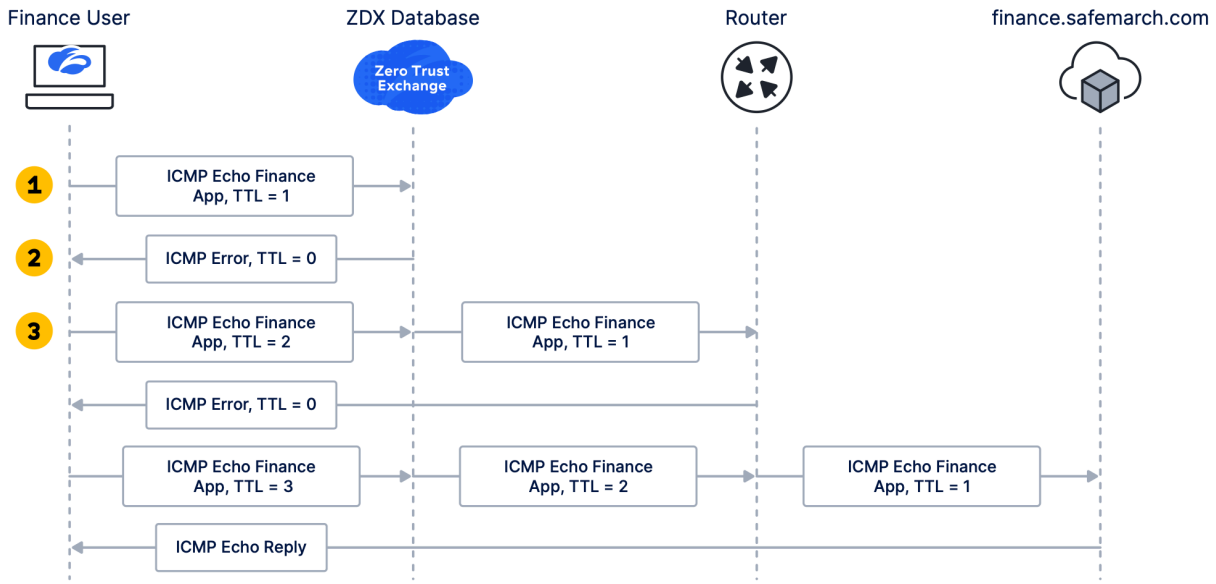
⚠️ Warning

Probe success is not user login success

- A Web Probe can validate that an application front door responds as expected. It does not necessarily prove that a user can complete every downstream authentication, authorization, or application workflow. Do not inflate a probe result into a full transaction test unless the configured probe actually measures that behavior.
- Web Probe errors should be interpreted according to the failure stage. Invalid destination URLs, unresolved domains, redirect loops, response codes outside the expected range, and request timeouts point to different remediation paths. The engineer should first determine whether the failure reflects the probe configuration, the destination application, DNS resolution, policy enforcement, or a transient service condition before escalating. Treat the error text as a breadcrumb, not a verdict from Mount Sinai.
- Web Probes become most useful when correlated with application dashboards and user details. A Page Fetch Time spike across many users and regions suggests a very different response than the same spike for one user on one device. The engineering move is to compare the application-level trend, the affected regions or departments, and user-specific web metrics before deciding whether to contain, escalate, or tune the probe.

Cloud Path Probes

A Cloud Path Probe measures the network path between the monitored source and the application or destination. Where a Web Probe answers “did the application endpoint respond as expected,” a Cloud Path Probe answers “what happened along the path to get there.” That difference is critical in scenario-based troubleshooting because a poor user experience can be caused by a healthy application reached through a degraded path, or by an unhealthy application reached through a clean path.

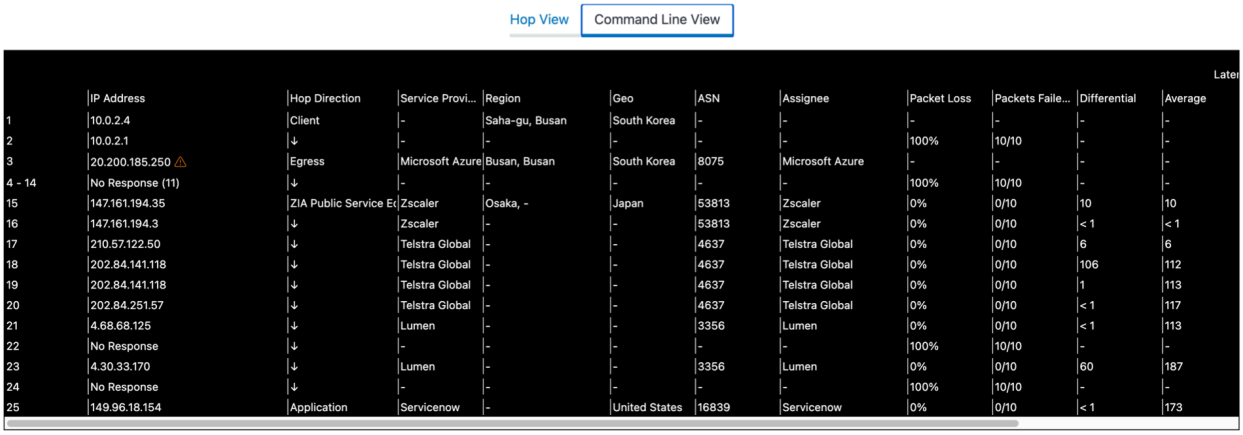


Cloud Path Probes collect path-oriented metrics such as hop count, packet loss, and latency. Latency is presented with values such as average, minimum, maximum, and standard deviation, which helps an administrator distinguish a consistently slow path from a path with unstable variation. Packet loss is shown by hop or path segment, but it must be interpreted carefully because not every intermediate hop treats probe traffic the same way it treats forwarded application traffic.

The practical value of Cloud Path is visibility across path segments. For internet and SaaS applications, the path might be direct from the endpoint through local egress to the destination, or it might traverse a ZIA Public Service Edge before reaching the application. For private applications, the path can include ZPA Service Edge and App Connector-related segments. A candidate should be able to recognize that the same symptom, such as high latency, means different things depending on which segment is affected.

Cloud Path Probes use traceroute-style path discovery, but ZDX presents the results in operational views rather than leaving the administrator to assemble raw command output by hand. Hop View gives a visual representation of the path, while Command Line View provides tabular detail that resembles traditional path diagnostics. Together, they help identify whether

the performance issue appears close to the endpoint, in the last mile, across an intermediate provider path, around a Service Edge segment, or near the destination.



	IP Address	Hop Direction	Service Provi...	Region	Geo	ASN	Assignee	Packet Loss	Packets Faile...	Differential	Average	Later
1	10.0.2.4	Client	-	Saha-gu, Busan	South Korea	-	-	-	-	-	-	
2	10.0.2.1	↓	-	-	-	-	-	100%	10/10	-	-	
3	20.200.185.250 ⚠	Egress	Microsoft Azure	Busan, Busan	South Korea	8075	Microsoft Azure	-	-	-	-	
4 - 14	No Response (11)	↓	-	-	-	-	-	100%	10/10	-	-	
15	147.161.194.35	ZIA Public Service E	Zscaler	Osaka, -	Japan	53813	Zscaler	0%	0/10	10	10	
16	147.161.194.3	↓	Zscaler	-	-	53813	Zscaler	0%	0/10	< 1	< 1	
17	210.57.122.50	↓	Telstra Global	-	-	4637	Telstra Global	0%	0/10	6	6	
18	202.84.141.118	↓	Telstra Global	-	-	4637	Telstra Global	0%	0/10	106	112	
19	202.84.141.118	↓	Telstra Global	-	-	4637	Telstra Global	0%	0/10	1	113	
20	202.84.251.57	↓	Telstra Global	-	-	4637	Telstra Global	0%	0/10	< 1	117	
21	4.68.68.125	↓	Lumen	-	-	3356	Lumen	0%	0/10	< 1	113	
22	No Response	↓	-	-	-	-	-	100%	10/10	-	-	
23	4.30.33.170	↓	Lumen	-	-	3356	Lumen	0%	0/10	60	187	
24	No Response	↓	-	-	-	-	-	100%	10/10	-	-	
25	149.96.18.154	Application	Servicenow	-	United States	16839	Servicenow	0%	0/10	< 1	173	

When reading Cloud Path data, the destination and downstream pattern matter more than a single scary intermediate hop. Routers and transit devices can rate-limit or deprioritize probe responses, especially for diagnostic traffic. If an intermediate hop shows loss but subsequent hops and the destination do not show continuing loss, the result may indicate probe-response handling rather than true end-to-end user-impacting packet loss. Do not turn one noisy hop into a sev-one unless the rest of the evidence agrees.

🎓 Exam Note

Intermediate hop interpretation

- Packet loss at an intermediate hop is not automatically root cause. Look for loss or latency that persists beyond that hop, appears at the destination, correlates with Page Fetch Time or availability degradation, and matches the affected user or location scope.
- Cloud Path Probes can follow a Web Probe when the network-path target should align to the same destination resolved by the web test. This is useful because it keeps the application transaction and the path diagnostic tied to the same monitored service. If the Cloud Path target is configured independently, the administrator must confirm that the host is representative. A beautifully measured path to the wrong destination is still wrong; it just looks professional while wasting time.
- Protocol selection is another design consideration. Cloud Path Probes can use protocols such as ICMP, TCP, UDP, or Adaptive mode where available. Adaptive mode is often a strong starting point because it can determine the most useful protocol behavior per path segment based on reachability and performance characteristics such as latency and loss. The administrator should still understand the application and network environment because firewalls, middleboxes, and providers may treat probe protocols differently.
- Packet count, interval, and timeout settings influence the quality and cost of Cloud Path data. More probe packets can improve confidence in loss and latency measurements, but they also increase probe activity. Shorter timing can improve detection sensitivity, but can also increase noise if the path or destination is variable.

The administrator-level decision is to keep settings conservative unless the monitoring objective requires additional granularity.

- For ZPA application scenarios, Cloud Path interpretation must include the private access architecture. The administrator should expect visibility that reflects the path from the user device toward ZPA infrastructure and then toward the App Connector and application, depending on how the traffic is carried. A private application with poor experience might have a local network issue, a Service Edge path issue, an App Connector-side path issue, or an application-side issue. Cloud Path helps narrow that field, but it does not replace understanding the ZPA design.
- Cloud Path is also important in UCaaS monitoring, but UCaaS paths can be dynamic. A meeting may use media destinations that change based on session placement, region, or provider logic. In supported call quality monitoring scenarios, Autosense Cloud Path behavior can provide a more accurate view because it detects and probes the active destination associated with the session instead of relying only on a fixed synthetic destination.

⚠️ Warning

Do not use Cloud Path as an application-health shortcut

- Cloud Path can show that the route to a destination looks healthy. It cannot prove that the web server returned the expected HTTP response, that a tenant login page is working, or that a SaaS workflow completed successfully. Pair Cloud Path evidence with Web Probe, application, user, and device telemetry before assigning root cause.
- In monitoring workflows, Cloud Path is most effective when correlated with Web Probe and device data. If Page Fetch Time is high and Cloud Path latency rises in the same time window, path degradation becomes more plausible. If Page Fetch Time is high but Cloud Path is clean, the investigation may shift toward application response, DNS behavior, or server-side conditions. If Cloud Path is degraded for only one user while device health shows Wi-Fi signal or CPU issues, the problem may be local rather than regional.
- Cloud Path errors should be handled according to the error type and scope. An invalid or unresolvable host points to target or DNS validation. Timeout or unreachable destination behavior may indicate path, destination, policy, or protocol handling. If errors appear across many users, locations, or hosted sources, the administrator should expand scope and compare application and incident views. If errors are isolated to one device, start closer to the endpoint before waking up half the network team.
- The best Cloud Path configuration is intentionally boring: correct target, correct scope, appropriate protocol behavior, reasonable timing, and clear correlation with the application it supports. It should produce evidence that helps an operator decide where to go next. If a Cloud Path Probe cannot help distinguish local access, ISP, Service Edge path, private access path, or destination-side conditions, it should be redesigned rather than admired.

Hosted Probes

Hosted Probes extend ZDX monitoring by changing the source of the test. Instead of running from the end-user device through Zscaler Client Connector, Hosted Monitoring runs from Zscaler cloud infrastructure as a multi-tenant service. That gives engineers an additional vantage point for monitoring a service or application directly from selected Zscaler-hosted locations.

The design value of Hosted Probes is independent confirmation. End-user probes tell you what users experience from their devices, networks, ISPs, and forwarding paths. Hosted Probes help show whether a service is reachable and performant from Zscaler cloud locations. When the two disagree, the difference itself becomes evidence: user-side issues, last-mile conditions, or policy path problems may be more likely when hosted results are healthy but user results are poor.

Hosted Monitoring organizes probes into collections. Collections let the engineer group Hosted Web and Hosted Cloud Path probes into independent monitoring sets. That object model matters because collection status, active probes, probe count, and hosted locations affect what data is generated. Disabling a collection is not a cosmetic choice; it suppresses the probes in that collection and changes the monitoring coverage.

Hosted Web Probes use many familiar Web Probe concepts: destination URL or IP address, request headers, expected HTTP response status, timeout, run frequency, redirect handling, and selected hosted locations. Additional options such as HTTP version and DNS override settings allow the engineer to validate a service using a specific DNS behavior when needed. These settings are powerful because they can isolate whether the service behaves differently under alternate name-resolution assumptions.

Hosted Cloud Path Probes focus on network performance from the hosted monitoring environment. They are used to evaluate path characteristics such as latency, packet loss, and jitter from the selected hosted locations toward the monitored destination. For scenarios involving regional service reachability or suspected provider path issues, hosted network probe data can complement user-side Cloud Path evidence without requiring a user device to reproduce the condition.

Companion probes introduce an important lifecycle rule. When a Hosted Web Probe is paired with a Cloud Path companion probe, the companion can provide network-path context for the web result. However, companion probes count toward Hosted Monitoring probe usage, and disabling or deleting parent and companion probes has operational consequences. Engineers should avoid treating companion probes as free decoration. They are useful, but they still consume monitoring capacity and add scope that must be governed.

Hosted Monitoring can also intersect with alerting. Alert templates can be applied to Hosted Probes so that recurring criteria do not have to be rebuilt repeatedly. This is useful when several hosted probes should share severity, threshold, and delivery expectations. The operational

guardrail is the same as any alert design: start with a known failure mode, define a threshold that represents user or service impact, and tune the criteria to avoid alert fatigue.

🎓 Exam Note

Hosted vantage point

- Hosted Monitoring is not a replacement for end-user monitoring. It answers a different question. Use it when the scenario requires a Zscaler-hosted perspective on a service or application, especially when you need to separate service availability from user-local, endpoint, or last-mile variables.
- Hosted Probe errors should be handled by separating configuration problems from platform or destination problems. Invalid destination values, maximum redirect failures, unsupported protocol selections, DNS resolution failures, and unreachable destinations lead to different next steps. If the error indicates internal processing, resource, or hosted probe system behavior, the correct escalation path may be Zscaler Support rather than endless local tuning. That is not giving up; that is routing the ticket to the right queue before it becomes archaeology.

When to Use Each Probe Type

Use a Web Probe when the requirement is to determine whether a web application endpoint is reachable, responsive, and returning expected web behavior. This is the right tool when a scenario describes slow page access, unexpected availability drops, DNS resolution delay for a web target, server response delay, or a need to validate a custom application URL.

Use a Cloud Path Probe when the requirement is to understand network path behavior. Cloud Path telemetry helps expose hop count, packet loss, latency, and path segments between the user device and the destination. When Page Fetch Time rises and the administrator needs to know whether the network path is responsible, Cloud Path is the evidence source that can separate local, ISP, Service Edge, and destination-side path behavior.

Cloud Path interpretation depends on the forwarding model. A direct internet path differs from a path through a ZIA Public Service Edge. A private application path through ZPA introduces ZPA Service Edge and App Connector-related visibility. The administrator should not force every scenario into the same mental diagram. The right interpretation starts with knowing how the traffic is expected to reach the application.

Network applications are an important exception to the “most applications need a Web Probe” pattern. A Network application can be monitored with a Cloud Path Probe because the monitoring objective is path visibility rather than web availability. In scenario terms, requiring a Web Probe for a non-web network monitoring objective is a great way to look industrious while measuring the wrong thing.

Autosense Cloud Path behavior applies when the monitored application destination can change dynamically, especially in supported UCaaS call quality scenarios. A normal Cloud Path Probe aimed at a fixed destination may not reflect the actual media path used during a meeting.

Autosense-style behavior is valuable because it can align path telemetry with the destination selected during the real communication session.

Use Hosted Probes when the monitoring requirement is independent of the end-user device. This is common when an organization wants continuous visibility from selected Zscaler-hosted locations, or when operations needs to compare user-side telemetry with a cloud-based monitoring source. Hosted Monitoring is especially useful when the scenario asks what the service looks like from Zscaler infrastructure rather than what one user sees from home, branch, hotel, or airport Wi-Fi.

🔄 Cross-policy resources

ZIA, ZPA, Client Connector, and Service Edge path

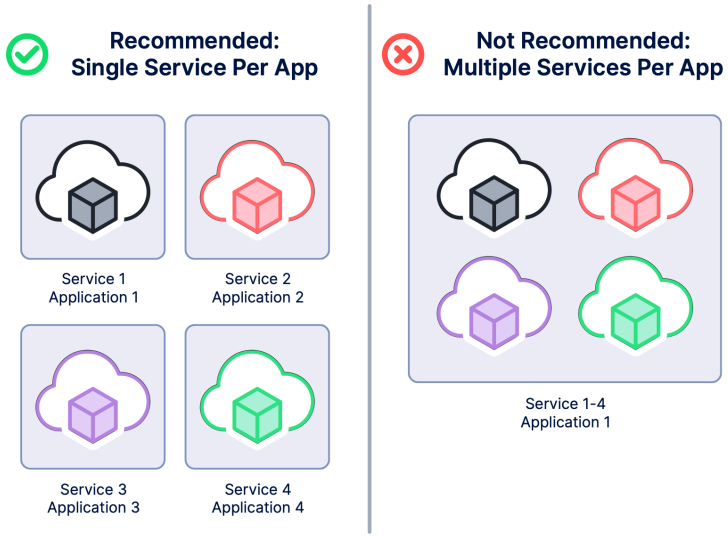
Probe evidence must be interpreted through the access architecture. ZIA forwarding, ZPA application access, Client Connector state, Service Edge selection, App Connector placement, and local egress all affect what the telemetry can prove. A correct probe choice can still be misread if the path model is wrong.

The fastest way to choose correctly is to eliminate what each probe cannot prove. A Web Probe cannot identify the hop where packet loss begins. A Cloud Path Probe cannot prove that an application returned the expected web response. A Hosted Probe cannot reproduce a user's endpoint CPU spike, Wi-Fi signal, local DNS behavior, or last-mile ISP condition. Each probe is useful because it is specific.

In many real scenarios, the answer is not one probe type but a sequence of evidence. A Web Probe may show that an application is slow. A Cloud Path Probe may show that the path is clean or degraded. Hosted Monitoring may show whether the service is healthy from a Zscaler-hosted location. The administrator's job is to combine these views without pretending any single view has magical root-cause superpowers.

Probe Configuration Best Practices

Start with an application monitoring strategy. Organization-wide applications should be monitored broadly because many users depend on them. Group-specific applications should be scoped to the teams that use them. Noisy, newly deployed, or business-critical applications may deserve more focused monitoring during rollout or stabilization. The design should reflect operational value, not just the order in which applications appeared in a meeting.



Keep each monitored application aligned to a single service or service instance whenever possible. Combining unrelated services, multiple regional front doors, or several application functions under one monitored application can distort scores and complicate troubleshooting. If the finance service has distinct regional endpoints, separate monitoring can prevent a regional problem from being averaged into an ambiguous global application story.

Use Probing Criteria and Exclusion Criteria as telemetry scoping controls. Inclusion criteria narrow the population across selected fields, while multiple selected values within a field can broaden the match within that field. Exclusion criteria remove selected users, groups, departments, locations, or devices from probe generation. Exclusions are evaluated before inclusions, so combined logic must be reviewed carefully.

Collection-level criteria are especially important. If criteria are defined at the collection level, they take precedence over probe-level criteria for that collection. This can surprise administrators who edit a probe and expect a different scope, only to discover that the broader collection logic is still controlling which devices generate telemetry.

For private applications, avoid over-scoping. Probe only the users, user groups, and departments that use the application unless there is a clear operational reason to do otherwise. Over-scoping private application probes can increase noise, waste probe capacity, and create misleading data from users who should never be part of the application's monitoring population.

Probe frequency should be chosen with subscription limits, application criticality, and operational need in mind. Higher-frequency monitoring can improve time-to-detect, but it can also consume capacity and increase monitoring noise. The right design balances detection speed against probe limits, active-probe counts, and the risk of unnecessary traffic against the monitored service.

Adaptive mode is often the right starting point for Cloud Path Probes because it can select the best available protocol behavior by path segment. The decision is based on path performance factors such as latency and loss. If adaptive discovery cannot complete as expected, fallback behavior is used. Administrators should understand that protocol choice affects visibility, especially when networks treat probe traffic differently.

🎓 Exam Tip

Defaults are safe, not sacred

- Use default settings when they produce useful telemetry. Change settings when the application's response behavior, redirects, DNS behavior, protocol handling, or monitoring scope requires a more accurate configuration. The exam usually rewards why the setting is changed, not the fact that a setting exists.
- Response codes and redirect behavior should be reviewed before the probe is trusted. A redirect can be a healthy application front door, an unnecessary extra dependency, or a misleading signal depending on the target. Maximum redirects should match normal application behavior. Unexpected response codes should

trigger configuration review before anyone declares an outage.

- After probe changes are made, validate activation and telemetry. Saving a change is not the same as making it effective across the ZDX control plane. Configuration updates must be activated so they can propagate. After activation, confirm that the expected users or hosted locations are producing data and that the relevant application views show probe status and metrics.
- Dashboard validation should include both application-level and user-level views. At the application level, review score trend, Page Fetch Time, affected regions, impacted departments, and probe status. At the user level, correlate Web Probe metrics, Cloud Path data, and device health. This prevents a common failure pattern: fixing the probe definition while leaving the real user experience unresolved.
- Governance is part of probe design. Use RBAC so that service desk users, monitoring administrators, and platform engineers receive only the access they need. A role that can view dashboards does not necessarily need probe-editing permissions. A role that can run diagnostics does not necessarily need broad configuration rights. Least privilege applies to monitoring operations just as much as it applies to security policy.

⚠️ Warning

Over-scoped monitoring can hide the truth

- Assigning every probe to every user may feel comprehensive, but it can dilute analysis, consume capacity, and create irrelevant telemetry.
- Scope probes to the population that actually uses the application and to the evidence needed for triage.
- Troubleshooting probe configuration should follow a layered review.
- Confirm the application is enabled, the probe is enabled, the target is valid, criteria include the expected users or locations, exclusions do not remove them, DNS resolves as expected, response behavior matches the configured success criteria, and path telemetry supports the suspected failure domain. That sequence beats random dashboard clicking every time.
- The final optimization habit is to revisit probe design after incidents. If a probe did not help isolate scope, identify the failure domain, or guide escalation, it should be tuned. If a probe generated alerts without action, refine the alert or the monitoring target. ZDX is most effective when probe configuration evolves with the application estate instead of fossilizing after onboarding.

📝 Quick Review

- Web Probes measure web application availability and responsiveness using metrics such as Page Fetch Time, DNS time, server response time, and availability.
- Cloud Path Probes measure network-path behavior such as hop count, latency, packet loss, and path segmentation.
- Hosted Probes run from Zscaler cloud infrastructure and are used when the scenario requires a hosted vantage point rather than an end-user device perspective.
- Autosense Cloud Path behavior is useful for supported UCaaS scenarios where the media destination can change dynamically.
- Predefined applications simplify setup, while custom applications require deliberate target, response, and scope design.
- Keep monitored applications aligned to one service or service instance whenever possible.
- Use Probing Criteria and Exclusion Criteria carefully; exclusions are evaluated before inclusions.
- For private applications through ZPA, scope probes to the users, groups, and departments that use the application.
- Review response codes, redirects, DNS behavior, and probe status before trusting the probe as production evidence.
- Avoid common mistakes: using Web Probes as full transaction tests, using Cloud Path Probes to prove web response behavior, over-scoping probe populations, ignoring collection-level criteria, and treating Hosted Monitoring as a replacement for end-user monitoring.

🏅 Exam Blueprint Alignment

Scenario Review

- Given a set of requirements, configure predefined applications and custom applications for meaningful ZDX monitoring.
- Given a set of requirements, configure managed monitoring and Hosted Monitoring when the requirement calls for a Zscaler-hosted vantage point.
- Given a set of requirements, monitor real-time application performance from a user perspective using browser-oriented performance metrics.
- Define and differentiate Web Probes, Cloud Path Probes, and Hosted Probes.
- Given a scenario, choose a Web Probe when the required evidence is application availability, Page Fetch Time, DNS time, server response time, or expected web response behavior.
- Given a scenario, choose a Cloud Path Probe when the required evidence is path visualization, latency, packet loss, hop count, or network bottleneck location.
- Given a scenario, choose Hosted Probes when the required evidence must come from Zscaler cloud infrastructure rather than an end-user device.
- Given a scenario involving private applications, account for ZPA path behavior, App Connector path visibility, and appropriate probe scoping.
- Given a scenario involving supported UCaaS monitoring, recognize when dynamic media-path behavior makes Autosense-style Cloud Path monitoring more appropriate

than a fixed destination probe.
- Given a scenario, use probing criteria, exclusion criteria, and collection scope to target the correct monitoring population while avoiding unnecessary probe volume.

Remediation and Optimization Scripts

Remediation and optimization scripts allow administrators to run approved PowerShell-based actions on selected devices. They can be used to fix known endpoint conditions, gather additional evidence, or perform maintenance tasks that support digital experience improvement. The feature is most valuable when it is tied to a precise operational hypothesis, such as “these affected Windows devices show a repeatable local condition that can be safely corrected or measured.”

The exam blueprint maps this subsection to scenario-based execution of approved remediation or optimization scripts at scale. “At scale” does not mean “all devices.” It means the administrator can safely translate a ZDX finding into a controlled job that targets the correct devices, respects prerequisites and permissions, and produces evidence that the action completed or failed for understandable reasons.

A script is the reusable operational object. A remediation job is the execution instance. The script defines what can be run, including its name, description, operating system, file, parameters, notification behavior, status, and privilege setting. The job defines who or what receives the action, when it runs, how long it may run, how long it waits for offline devices, and how results are tracked.

Candidates should avoid thinking of scripts as generic endpoint automation. ZDX remediation is governed through ZDX-specific controls: Script Management, Script Templates, Remediation Jobs, Remediation Settings, device compatibility checks, script signing, optional end-user confirmation, and job tables for In Progress, Completed, and Scheduled work. Those controls are part of the answer in real scenarios, not background decoration.

🎓 Exam Note

Script execution is a governed workflow

A remediation action includes the script object, administrator permission, target selection, device eligibility, execution settings, notification behavior, result tracking, and outcome validation. In scenario questions, prefer the option that preserves these controls over the option that simply runs the broadest action fastest.

Overview of ZDX Remediation Scripts

ZDX remediation scripts help administrators address device-side issues without requiring each user to walk through manual support steps. The feature supports proactive operations: when a recurring endpoint condition is detected, an approved script can be run against selected devices to correct the issue or collect additional data for further analysis.

The operational value is strongest when the script is tied to evidence from ZDX. Device Health, User Details, software crash data, system hangs, CPU utilization, memory usage, disk usage,

Wi-Fi signal quality, and device events can all help determine whether a script is appropriate. If the evidence points to application availability or an upstream network path, a device-side remediation script may be the wrong action even if it is available.

Scripts can be predefined or custom. Predefined scripts are certified and ready for use, which reduces preparation overhead and is useful for common operational tasks. Custom scripts provide flexibility for organization-specific troubleshooting or maintenance, but they require stronger governance because the organization is responsible for validating the script, signing it, distributing trust as needed, and testing it before production use.

A remediation job runs a selected script against selected targets. The target selection can be based on devices, users, user groups, locations, location groups, or departments. This selection model is important because it allows the administrator to align the job with the actual impact scope shown in ZDX rather than guessing. If only a department, location, or user group is affected, the job scope should match that evidence.

Scripts may also include parameters. Parameters let a script adapt to different approved use cases without requiring multiple near-duplicate script uploads. Mandatory parameters help enforce required input, while example values guide the administrator who runs the job. A reusable script with clean parameters is easier to govern than a stack of “final-final-v3” scripts hiding in the library like a 1998 shared drive.

Script output should be treated as part of the design. Custom scripts can route output and result status back into ZDX, enabling administrators to see what happened without logging into each endpoint. A good script reports what it checked, what it changed, and whether the expected condition was reached. Poor output forces a second troubleshooting cycle and weakens the value of automation.

The most effective administrators do not start with the question, “Which script can I run?” They start with, “What condition am I trying to prove or change?” That distinction matters. One workflow may call for a low-impact data-collection script; another may call for an elevated maintenance script; a third may call for no script at all because the evidence points outside the endpoint.

💬 Sidebar
Baseline-to-ZDXA bridge

At the administrator baseline, candidates learn where ZDX shows user and device telemetry. At the ZDXA level, candidates must use that telemetry to decide whether an endpoint action is justified, how narrowly it should be scoped, which permissions are required, and how success will be measured.

Uploading and Managing ScriptsExecuting Scripts at Scale

Uploading a script creates an operational object that can be reused in remediation jobs. The script definition includes a name, description, operating system, uploaded file, optional parameters, optional end-user confirmation, and whether the script runs with elevated permissions under the system account. These fields should be completed as if another administrator will need to understand and run the script during an active incident.

The uploaded script file is a PowerShell script, and the uploaded source material identifies a maximum script file size of 512 KB. That limit reinforces a good design principle: remediation scripts should be focused, testable, and purposeful. If a script becomes large enough to behave like an endpoint management platform inside a script file, it probably needs to be redesigned.

Custom scripts must be signed before they are uploaded and run. Signing is not busywork; it is the trust boundary for remote script execution. The certificate must be valid, trusted appropriately on target devices, and not expired or revoked. Scripts should also be encoded using accepted formats, such as ASCII or UTF-8 without a byte order mark, because unsupported encoding can prevent successful processing.

The elevated-permissions option should be used only when the task requires it. Running as the system account may be necessary for device-level maintenance, but it increases operational risk. If a script can accomplish the task in a user context, that is usually the cleaner least-privilege design. In scenario terms, “works with elevated permissions” is not the same as “should be elevated.”

Script parameters should be designed for safe reuse. Mandatory parameters should be used when the script cannot safely run without a value. Example values should be clear enough to prevent misinterpretation. Parameter naming should be operationally meaningful, especially when scripts are used by teams that did not author them.

End-user confirmation provides a user-awareness control before a script runs. If the script may interrupt work, affect the user session, collect information the user should know about, or make a visible change, showing a notification may be appropriate. Silent execution is best reserved for low-impact, approved maintenance or collection actions where user interaction is unnecessary and organizational policy allows it.

The script library also requires lifecycle management. Administrators can view, run, edit, disable, enable, or delete scripts, depending on permissions and status. If a script is disabled, it cannot be run until it is enabled. Disabling is useful when a script is being reviewed, retired, replaced, or temporarily blocked because of a defect or policy concern.

Predefined scripts can be viewed, searched, and imported from script templates. They are useful because they are certified and reduce the overhead associated with custom signing. However, predefined does not mean universally safe for every environment. They should still be

tested in a non-production environment and reviewed against local endpoint standards before broad deployment.

> ⚠️ **Warning**
> *A reusable script is still a change*
>
> A script that ran successfully yesterday can still be risky today if the target population, endpoint version, certificate state, execution policy, or business window changed. Treat each broad run as a new operational event, not a replay button.

Prerequisites and Permissions

Before a remediation workflow can succeed, the tenant must support the feature. The organization needs the appropriate ZDX subscription level, and remediation must be enabled in the administrative configuration. If the entitlement or remediation setting is missing, an administrator may be able to view ZDX telemetry but still be unable to run scripts against devices.

Role permissions are equally important. Script Management allows administrators to configure or view scripts. Script Templates allows importing or viewing predefined templates. Remediation Jobs allows running or viewing jobs. Remediation Settings allows configuring or viewing the settings that govern remediation. A least-privilege role design separates these permissions according to operational responsibility.

A service desk role might only need to view user context, run tightly approved jobs, or review job results. A platform administrator might need to upload scripts, configure settings, and manage templates. Combining all remediation privileges into one broad role may be convenient, but it weakens accountability and makes audit review harder. RBAC is the difference between controlled automation and “who gave the intern a chainsaw?”

Target devices must meet version compatibility and operating system requirements. If a selected device is not compatible, the script does not run on that device. This is an expected platform safeguard. In a mixed fleet, a job may complete with only a subset of devices successfully processed, so administrators must interpret success rates and device-level results instead of assuming uniform execution.

The uploaded remediation source identifies Windows-focused script execution requirements, including minimum Client Connector and ZDX module compatibility for Windows devices. It also identifies endpoint preparation requirements such as ensuring PowerShell execution policy is not restricted and allowing required script execution components according to Windows version. These endpoint prerequisites are often the real issue when a script appears valid but cannot execute.

Custom script signing requires both a trusted certificate path and a valid code-signing certificate. The signing certificate must be trusted by the devices that will run the script. Certificate trust

commonly involves Trusted Root Certification Authorities and Trusted Publishers on Windows devices. A script signed by a certificate that target devices do not trust is still an operational failure.

PowerShell execution policy can block remediation. If the endpoint is in a restricted execution mode, script execution may be disallowed or limited to signed scripts. In those scenarios, the correct response is not to weaken endpoint controls casually. The correct response is to align the script, signing model, and endpoint policy with the organization's security standard.

🎓 Exam Tip

Diagnose prerequisites before blaming the script

When a script does not run, check subscription support, remediation enablement, admin permission, script status, target compatibility, operating system support, Client Connector and ZDX module version, certificate trust, and PowerShell policy. A failed run is often a readiness problem, not a code problem.

Safety and Change Control Considerations

The first safety control is scope. ZDX allows target selection by devices, users, user groups, locations, location groups, and departments. Administrators should select the narrowest population that matches the evidence. If a problem is isolated to a department or location, an all-users remediation job is usually unnecessary and may create avoidable operational noise.

The second safety control is script intent. A data-collection script, a user-context maintenance script, and an elevated system-level change script are not equivalent. Each requires different review, notification, scheduling, and validation expectations. A low-risk collection job might run during business hours, while a user-impacting cleanup or service action might require a maintenance window.

Change control starts before the job is created. The script should be reviewed, tested, signed, and documented. The job should have a clear name that ties it to the incident, request, change record, or operational purpose. If someone reviews the job later, the name and output should explain why it was run without needing a scavenger hunt through chat history.

Execution settings should match the expected behavior of the script. Administrators define how long the script can run for the job, and they can specify how long the job waits for devices to come online. A short command should not receive an unnecessarily long runtime, and a long-running maintenance action should not be configured with a timeout that guarantees failure. The settings are part of operational safety.

Scheduling can reduce user impact and improve success. A job can be scheduled for later when the action should occur during a controlled window. For global organizations, scheduling requires more than selecting a convenient local time. The administrator should consider regional business hours, device availability, and the urgency of the incident.

End-user notification should be aligned to impact and policy. A notification can identify the sending team, provide a header, explain the action, and define the accept button text. This is useful when user awareness or confirmation is appropriate. Hiding notifications may be valid for low-impact approved actions, but it should not be used to avoid explaining a user-visible change.

Common failure patterns should be understood before production use. Jobs can fail or be aborted because of expiration, administrative abort, remediation disabled for a user or device, unsupported task category, missing policy configuration, user deferral, maximum retry or defer limits, download failure, signature validation failure, unsigned scripts, revoked certificates, execution policy blocks, runtime errors, handler initialization failure, or unavailable orchestration services. Each category points to a different corrective action.

A safe operational process uses staged deployment. Start with a small representative test set, validate output and device behavior, then expand to the affected population. For a severe incident, the stages may be compressed, but they should not disappear. The fastest way to make a device problem exciting is to automate it across the enterprise.

🔄 Cross-policy resources

Endpoint and governance intersections

Remediation decisions intersect with Client Connector readiness, endpoint security policy, PowerShell execution policy, certificate trust, RBAC, ITSM records, and audit expectations. The ZDX administrator does not need to own every adjacent control, but must know when a failure belongs to endpoint readiness, script trust, permission design, or change governance rather than ZDX telemetry.

Validating Remediation Outcomes

Validation starts on the Remediation Jobs page, where jobs are grouped into In Progress, Completed, and Scheduled tables. These views allow administrators to determine whether a job is actively running, finished, or waiting for a scheduled time. Status is the first validation step, but it is not the same as proving the user experience improved.

For an in-progress job, review progress, selected device count, script name, creator, and start time. This helps determine whether the job is advancing normally or whether it is stalled. If the job is waiting for devices to come online, progress must be interpreted in context. An offline target is not the same as a script defect.

For completed jobs, the success rate is a key indicator. The Completed view can show how many selected devices completed successfully and how many did not. Administrators should drill into job details to review impacted devices, execution status, error messages, result values, and script output. This is where the job becomes evidence rather than hope with a progress bar.

The Scheduled view is also part of validation. Scheduled jobs can be reviewed and, where permitted, edited or deleted before execution. This matters when a scope mistake is discovered, a business window changes, or new information shows that the action is no longer appropriate. Governance includes preventing the wrong job from running, not just reacting after it does.

Script output should be interpreted against the intended result. If the script was designed to collect diagnostics, success means the expected information was captured and is usable. If the script was designed to remediate a condition, success means the script reached the expected state and returned a meaningful result. “No error” is helpful; “condition verified after change” is better.

Error interpretation should guide the next action. Expiration errors usually point to job timing. Remote execution disabled errors point to remediation settings for the user or device. Unsupported task or validation errors point to script structure or category support. Download failures point to network or URL issues. Signature, unsigned, revoked, or certificate validation errors point to signing and trust. Runtime errors point to script logic, while timeout errors may require runtime adjustment or script redesign.

Validation should return to the original ZDX telemetry. If the job was intended to reduce device CPU impact, review Device Health, CPU-related contributing factors, and the affected user’s post-run experience. If the script addressed a Wi-Fi-related issue or collected wireless state, correlate the output with Wi-Fi signal quality, network adapter details, and device events. If it addressed software instability, review crashes, hangs, or process-related indicators after execution.

At scale, validation also includes exceptions. The administrator should identify which devices succeeded, which failed, which were skipped, which were incompatible, and which require follow-up. Exporting device lists or results may be appropriate when a separate endpoint team, desktop engineering group, or service desk process must continue the work. The remediation job is often not the end of the incident; it is the evidence that tells the next team exactly where to focus.

The final operational decision is close, rerun, revise, or escalate. Close when the job succeeded and telemetry confirms improvement. Rerun when the failure was transient and the device is now available. Revise when parameters, signing, runtime, or logic were wrong. Escalate when the evidence shows that the issue is not endpoint-local or when the failure indicates a platform, certificate, policy, or orchestration issue outside the administrator’s authority.

🎓 Exam Note

Completed does not always mean remediated

A completed job proves an execution outcome. It does not automatically prove the original ZDX issue is resolved. In scenario-based reasoning, validate both the remediation job result and the telemetry that justified the action in the first place.

📝 Quick Review

- Start with ZDX evidence, then decide whether a script is the correct action.
- Separate the script object from the remediation job: the script defines the reusable action; the job defines execution scope and timing.
- Use predefined scripts when they fit the use case, but still test them against your environment before broad deployment.
- Custom scripts must be signed, trusted, and prepared according to endpoint policy requirements.
- Keep script scope narrow by selecting the affected devices, users, user groups, locations, location groups, or departments.
- Confirm subscription support, remediation enablement, role permissions, script status, target compatibility, and endpoint readiness before troubleshooting script code.
- Use elevated execution only when the task requires system-level permissions.
- Use end-user confirmation when the action may be visible, disruptive, or policy-sensitive.
- Validate through job status, success rate, device-level results, script output, error messages, and the original ZDX telemetry.
- Common mistakes to avoid: running broad jobs too early, ignoring certificate trust, treating predefined scripts as automatically production-ready, using elevated permissions by default, overlooking offline-device wait settings, and assuming a completed job proves user-experience recovery.

🏅 Exam Blueprint Alignment

Scenario Review

- Section 2.3: Given a scenario, execute approved remediation or optimization scripts at scale and validate outcomes, including prerequisites, permissions, and safety considerations.
- Section 3.1: Analyze device telemetry such as CPU, memory, Wi-Fi, software crashes, and device health before deciding whether endpoint remediation is appropriate.
- Section 3.2: Analyze diagnostic or execution output to determine whether the remediation succeeded, failed, or requires revision.
- Section 4.1: Assign role-based access control so script management, script template access, remediation job execution, remediation settings, and view-only review are granted according to operational responsibility.
- Section 4.2: Use alerting or operational signals as triggers for investigation, but validate scope before using a script as the remediation action.

Browser-Based Application Performance Monitoring

Browser-based application performance monitoring focuses on how web applications behave from the user's perspective. In ZDX, this is primarily represented through synthetic Web Probes, Cloud Path visibility, application dashboards, user details, and Real User Monitoring when enabled. Candidates should understand how these signals complement each other rather than treating them as interchangeable.

Synthetic monitoring provides controlled, repeatable measurement. A Web Probe can evaluate whether a configured URL responds as expected, how long the page fetch takes, how DNS and server response behave, and whether the application is considered available. Real User Monitoring captures what happens during actual browser interactions on supported devices, adding insight into page load behavior and user-session experience that a synthetic probe may not fully represent.

The blueprint emphasis in this subsection is scenario driven: configure business-critical applications, select the correct monitoring type, interpret browser-based metrics, and validate end-user experience. The exam is not trying to turn candidates into dashboard tourists. It is testing whether they can reason from a requirement to a monitoring design, then from monitoring evidence to an operational action.

> 🎓 **Exam Note**
> *Synthetic and real-user evidence*
>
> Synthetic monitoring and Real User Monitoring answer different questions. Synthetic probes ask, "Does the configured target respond from this vantage point?" Real User Monitoring asks, "What did the browser session experience?" When the signals disagree, investigate the difference instead of forcing one to "win."

Monitoring Real-Time User Experience

Real-time user experience monitoring starts with the endpoint and the browser. ZDX uses Zscaler Client Connector to collect endpoint and probe telemetry, while Real User Monitoring captures browser-based interaction data for supported Windows and macOS devices. This combination allows administrators to evaluate user experience across the application, device, and network path instead of relying on a single availability signal.

RUM must be planned as part of the monitoring rollout. The organization needs the appropriate subscription, a role that can configure applications and view the relevant dashboards, compatible Client Connector and ZDX module versions, a supported browser with the RUM extension installed, and required endpoint processes permitted by local controls. If those prerequisites are missing, the issue is not a bad application score; it is incomplete instrumentation.

Use RUM for applications where browser behavior materially affects productivity. Identity portals, CRM systems, ITSM platforms, expense systems, and collaboration web apps are good examples because users care about whether the page actually becomes usable, not just whether a server returned a technically acceptable response. Synthetic probes may confirm reachability, while RUM can expose page load delays, blocking behavior, and browser-session errors.

The strongest operational interpretation comes from comparing synthetic and RUM signals in the same time window. If Web Probe metrics remain stable while RUM degrades, investigate browser behavior, endpoint resource pressure, application client-side errors, or session-specific content loading. If both synthetic and RUM signals degrade together for the same application and population, the administrator should look more closely at application responsiveness, DNS, availability, or the network path.

The Users Overview helps administrators move from aggregate experience to affected individuals. When applications are selected, the user distribution and user list show how users fall into experience categories, which users are impacted, and which devices are associated with them. This is especially important when a service desk ticket sounds isolated but the same pattern appears across several users or locations.

User Details provides the deeper evidence required for real-time validation. Device information, selected application scores, Web Probe metrics, RUM data when available, Cloud Path visibility, device health, and device events all belong in the same reasoning chain. A user who reports a slow application may actually have high CPU, weak Wi-Fi, DNS delay, an application-side response issue, or a path problem. ZDX helps reduce the guesswork.

Time selection is part of the diagnosis. The administrator should align the dashboard time range with the reported user experience rather than reviewing a convenient but irrelevant window. A clean graph after the complaint ended does not prove the issue never happened. A focused time range, compared against nearby normal behavior, gives the evidence the context it needs.

> ⚠️ **Warning**
> *Missing RUM data is not proof of application health*
>
> If RUM data is absent, validate the prerequisites before concluding that users had no browser-side issue. Check whether the extension is deployed, whether the device and browser are supported, whether the user actually accessed the application, and whether the administrator role can see the relevant data.

Browser-Based Performance Metrics

Page Fetch Time is the key synthetic metric for browser-based web application monitoring. It measures how long it takes to fetch the page specified in the Web Probe URL. Because the Web Probe requests the top-level page document rather than every embedded page object,

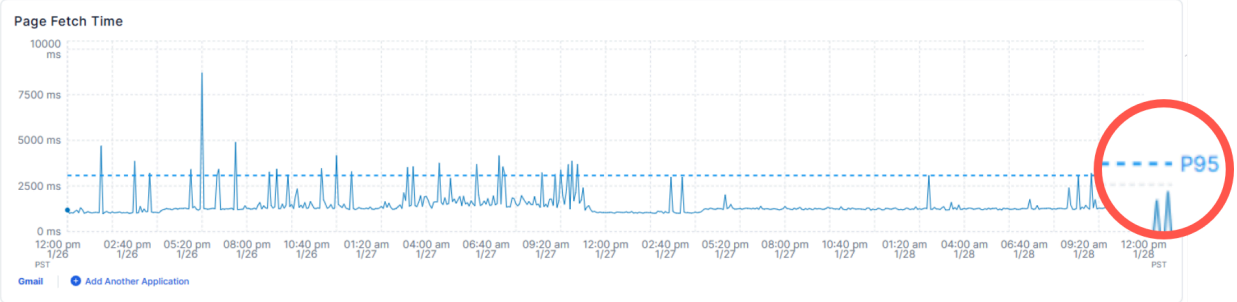
Page Fetch Time should be interpreted as a repeatable transaction measurement, not as a complete visual rendering measurement.

A Web Probe succeeds or fails based on the configured transaction behavior. For many web applications, a redirect to an identity provider or login flow may be expected and should be treated as a valid response if that is the intended success condition. For other applications, accepting broad response codes can hide real problems. The administrator's job is to define what "healthy" means for that application, not to assume every HTTP response is good news.

The browser-based transaction can be broken into several timing components. DNS time reflects name resolution. TCP connection time reflects session establishment at the transport layer. SSL or TLS handshake time reflects secure connection setup. Server Response Time, commonly associated with Time to First Byte, reflects how quickly the server begins responding. These measurements help determine whether slowness is caused before the server, at the server, or during the response.

ZDX Score provides a normalized way to summarize experience on a 0–100 scale. Good, Okay, and Poor ranges help administrators compare users, applications, and locations quickly, but the score still has to be interpreted against the underlying metrics. For browser-based applications, Page Fetch Time and availability are central contributors, and baseline behavior matters because an acceptable experience for one geography or application may not match another.

Percentile data helps separate average behavior from user pain. A Page Fetch Time average can look acceptable while a subset of users experiences poor performance. The P95 line shows the value under which most fetch times fall, making it useful for identifying outliers and understanding whether a few users or a broader population is affected. When P95 climbs while the average stays moderate, the tail of the experience is doing the talking.



RUM adds page-session metrics that synthetic Web Probes cannot fully represent. Page Load Time, Page Load Time and Blocking Time trends, and Errors Over Time help administrators understand whether real users experienced load delays, input responsiveness problems, JavaScript errors, or network errors while using the web application. These are browser-experience indicators, not just reachability indicators.

Metric interpretation should remain evidence-led. A DNS spike does not equal application failure. A high Server Response Time does not prove a local network issue. A clean Web Probe

does not rule out RUM problems. A poor RUM trend does not automatically mean the Service Edge path is unhealthy. Candidates should learn to let the metric sequence narrow the domain of responsibility before choosing the next operational step.

🎓 Exam Tip

Follow the transaction before blaming the path

For browser slowness, reason through DNS, connection setup, secure handshake, server response, redirects, transfer behavior, and real-user page load evidence. Do not jump straight to “network latency” unless the Cloud Path and timing data support it.

Business-Critical Application Monitoring

Business-critical application monitoring starts with application selection. Organization-wide services such as identity, collaboration, file sharing, CRM, ITSM, and productivity applications usually need broad visibility because their degradation affects many teams. Department-specific systems should be monitored for the users and locations that depend on them. New or noisy applications may need temporary focus because they create support demand during rollout or change windows.

Predefined applications are appropriate when the monitored SaaS application matches an available template. They reduce configuration effort because the application definition and default probes are already known after onboarding. Custom applications are appropriate when the organization needs to monitor an internal application, a private application, a unique SaaS tenant, or an externally hosted service that is not represented by a predefined application.

Most custom web applications require at least one Web Probe to be enabled for monitoring. Network applications are the exception because they do not require a web front end and can be monitored with a Cloud Path Probe. This distinction is an exam-friendly scenario pattern: if the requirement is application availability and browser responsiveness, think Web Probe; if the requirement is path visibility to a network destination, think Cloud Path; if both matter, use both.

A business-critical application object should represent a clean service boundary. Combining unrelated services, multiple regional application stacks, or several front doors into one monitored application can distort scores and make triage harder. If the finance application has separate regional endpoints, monitoring them as separate services may produce clearer evidence than blending all regions into one object and calling it “Finance.” That shortcut usually ages like milk.

Probe scope should be deliberate. Probing criteria and exclusion criteria can target user groups, users, departments, locations, location groups, and devices. The correct scope is the population that depends on the application and can generate meaningful evidence. Too broad a scope can consume capacity and create noise; too narrow a scope can miss the affected users when an incident occurs.

Hosted Monitoring provides an additional vantage point from Zscaler cloud locations. Hosted Web and Cloud Path probes can monitor a service from selected data centers, which is useful when administrators need a stable external view independent of a user device. It is not a replacement for endpoint-based monitoring; it is a complementary signal that helps determine whether an issue is visible from the cloud, from users, or both.

UCaaS applications require special care because media paths can be dynamic. Collaboration platforms may have a web application front door, signaling behavior, meeting quality metrics, and media transport paths that do not all use the same endpoint. For call quality scenarios, administrators should correlate meeting metrics with Cloud Path evidence and avoid assuming that a static web probe explains the live meeting experience.

Operational guardrails matter for business-critical monitoring. Subscription limits, active probe limits, per-user probe assignment, role permissions, and feature availability affect what can be configured and what administrators can see. A scenario that asks for monitoring design may be testing entitlement and scope discipline as much as technical probe knowledge. The cool kids call that governance; the rest of us call it “not breaking production by Friday.”

🔄 Cross-policy resources

ZDX, Client Connector, ZIA, ZPA

- Bring in Zscaler Client Connector when endpoint telemetry or probe execution matters.
- Bring in ZIA when the application path uses Internet and SaaS forwarding through a Public Service Edge.
- Bring in ZPA when the application is private and the path includes Private Access components.
- Do not expand into unrelated policy theory unless it changes the monitored path or the evidence.

Validating End-User Experience

Validation should begin at the broadest useful scope and narrow only when the evidence supports it. The Performance Dashboard helps identify the most impacted applications, affected regions, score trends, and Page Fetch Time behavior. When a report states that a specific SaaS application is slow in a country or region, the administrator should filter to that application and region, then identify whether the impact is concentrated or widespread.

The Applications Overview is the application-centered validation surface. It helps compare monitored applications, inspect score trends, locate impacted regions and departments, and review probe status. If one application is degraded across several regions, the likely domain differs from a scenario where many applications are degraded in one office or geography. Scope is not paperwork; scope is the difference between a useful escalation and a meeting with twelve people staring at each other.

The Users Overview and User Details views validate individual experience. When multiple applications are selected, candidates should remember that user score interpretation can be influenced by the poorest selected application. On the User Details page, the selected application updates the downstream evidence, including score trend, Web Probe metrics, Cloud Path, RUM data when available, device health, and device events.

Smooth ZDX Score helps reduce short-term variation and reveal the trend line, while the raw ZDX Score remains useful for point-in-time investigation. When a user reports a sharp performance drop, the administrator should inspect the actual time of the complaint, compare the smoothed trend to the raw score where appropriate, and then examine the web, path, and device metrics for that same moment.

Cloud Path validation is essential when the issue may involve the local network, ISP, Service Edge path, or application-side network segment. Hop View and Command Line View help administrators evaluate latency, packet loss, hop behavior, and the legs between user, egress, Service Edge, and destination. Intermediate-hop packet loss should be interpreted carefully because some routers rate-limit probe traffic; destination loss and correlated application symptoms carry more operational weight.

Root cause analysis and comparison features help turn evidence into a working hypothesis. When a Poor score appears, ZDX can analyze the selected point or range and return potential contributing factors with confidence. Comparing the affected point to a last known good score or another time window can highlight differences in Web Probe, device, and Cloud Path metrics. The administrator should use the result to guide investigation, not as a substitute for judgment.

Alerting and automation should support operational response without weakening least privilege. Alert rules, dynamic alerts, webhooks, ServiceNow integration, diagnostics sessions, and API workflows can all extend ZDX evidence into service management. The administrator must ensure that the role, integration, and action are appropriate: view-only for observers, diagnostic rights for service desk workflows, and configuration authority only for the administrators who need it.

Containment in ZDX scenarios is usually about narrowing blast radius and routing work to the right owner. If the evidence points to one user’s device, the action differs from a Last Mile ISP brownout, a regional application degradation, or a private application path issue. Good validation should answer four questions before escalation: who is impacted, which application is affected, where the issue appears in the path or transaction, and what changed compared with a known-good state.

For the exam, the safest reasoning pattern is evidence hierarchy. Start with application and region scope, validate affected users, inspect browser and Web Probe metrics, evaluate Cloud Path only where path evidence is relevant, correlate device health and events, and use automation or escalation only after the responsible domain is reasonably clear. That approach keeps the administrator from chasing ghosts, and ghosts are terrible at submitting packet captures.

⚠️ Warning

Common validation traps

Do not treat a good average score as proof that no users were impacted. Do not treat missing RUM data as proof of browser health. Do not assume intermediate-hop packet loss is the root cause without checking destination behavior and related metrics. Do not group unrelated services into one application and expect clean incident scope.

📝 Quick Review

- Operational configuration is the design of meaningful measurement, not just the activation of monitoring objects.
- Browser-based monitoring combines synthetic Web Probe evidence, Cloud Path visibility, dashboard interpretation, and RUM when enabled.
- RUM captures real browser-session experience and should be validated against prerequisites before troubleshooting missing data.
- Page Fetch Time measures the configured web transaction and should be interpreted with DNS, TCP, TLS, server response, redirect, and availability context.
- Define business-critical applications around clean service boundaries; avoid combining unrelated services or regional stacks into one monitored object.
- Use predefined applications when the SaaS template fits; use custom applications when the monitored service must be explicitly defined.
- Use Web Probes for browser application availability and responsiveness; use Cloud Path Probes for network path visibility; use Hosted Monitoring when a Zscaler cloud vantage point is required.
- Validate impact by moving from Performance Dashboard scope to application scope to user and device evidence.
- Common mistakes to avoid: accepting the wrong response codes, ignoring RUM prerequisites, over-scoping probes, misreading P95, overreacting to intermediate-hop loss, and escalating before comparing against a known-good state.

🏅 Exam Blueprint Alignment

Scenario Review

- Given a set of requirements, configure predefined applications, custom applications, hosted monitoring, and Real User Monitoring when the monitoring objective requires them.
- Given a scenario, differentiate between Web Probes, Cloud Path Probes, and Hosted Probes, and select the probe type that produces the necessary evidence.
- Given business-critical application requirements, configure monitoring from the end-user perspective using browser-based performance metrics.
- Given a report of regional application slowness, use the Performance Dashboard and application-level views to determine incident scope and affected users.
- Given conflicting synthetic and RUM evidence, compare Web Probe metrics, RUM metrics, Cloud Path metrics, and device health before selecting the likely domain of responsibility.

- Given a custom application design, define application boundaries and probe assignment so that dashboard output maps cleanly to service ownership and escalation paths.
- Given operational automation requirements, apply least-privilege thinking to alerting, diagnostics, API access, webhook delivery, and ITSM integration.

Troubleshooting and Root Cause Analysis (30%)

Analyzing Device, Network, and Application Telemetry

This section focuses on the most important troubleshooting skill in the domain: reading telemetry across tiers without over-weighting one data point. Endpoint telemetry explains whether the user's device is capable of delivering the experience. Network telemetry explains whether the path introduces latency, loss, hop instability, or ISP-related degradation. Application telemetry explains whether the service is available, responsive, and behaving as expected.

At the administrator baseline, candidates usually know that ZDX includes scores, dashboards, probes, and user details. At the ZDXA level, the expectation is stronger. You should be able to look at the same symptom from multiple perspectives and explain why one root cause is more likely than another. "The score is poor" is a symptom statement. "The affected users share a last-mile ISP, the destination loss aligns with the poor-score window, and unrelated applications show similar degradation" is troubleshooting analysis.

The ZDX Score is useful because it compresses complex experience data into an accessible signal, but it should never be interpreted alone. Strong analysis uses the score to locate the affected time window and then reviews the underlying contributors: endpoint health, Web probe metrics, Cloud Path behavior, incidents, inventory, and event correlation. The score gets you to the neighborhood; the telemetry tells you which house is on fire.

Time comparison is a recurring scenario pattern. Compare the impacted time to a known-good period, compare affected users to unaffected users, and compare the affected application against other applications. This approach keeps the administrator grounded in evidence instead of chasing every possible cause in sequence.

Endpoint Telemetry Analysis

Endpoint telemetry is the first place to rule out local causes that can mimic network or application problems. A slow browser, delayed collaboration client, or degraded application experience can be caused by the user's device even when the application and network path are healthy. Processor utilization, memory pressure, disk behavior, network I/O, battery state, Wi-Fi signal strength, and device events all provide context for the user's perceived experience.

The most important endpoint question is whether the device condition aligns with the affected time window. A brief processor spike ten minutes after the complaint is weak evidence. Sustained processor pressure, memory exhaustion, or disk contention during the same interval as the poor score is much stronger. For scenario reasoning, the administrator should connect

the metric to the symptom and then decide whether the issue is isolated to a device or repeated across a population.

Wi-Fi telemetry deserves special attention because it sits at the boundary between endpoint and network analysis. Weak signal strength, high access point latency, retransmission-related symptoms, or packet loss near the first hops can make a SaaS or private application appear unhealthy. If the issue follows an SSID, access point, or office area rather than a specific application, the investigation should shift toward wireless conditions instead of application escalation.

User Details can help determine whether the problem follows the user, the device, the location, or the application. If one user has multiple managed devices and only one device shows poor experience, the device hypothesis becomes stronger. If the same user has poor experience across devices only in one location, the local network or Wi-Fi hypothesis becomes stronger. If multiple users have the same experience against one application across different locations, the endpoint is less likely to be the primary cause.

Device Health views are more useful for population analysis than for one-off curiosity. If a department, model, operating system, or user group repeatedly shows poor device health, the administrator can identify whether the issue is operationally systemic. That supports more precise remediation, such as endpoint tuning, software correction, hardware refresh planning, or targeted user support.

Diagnostics sessions and deeper tracing are appropriate when dashboard telemetry narrows the issue but does not provide enough granularity. Use them to validate a hypothesis, not as the first click in every investigation. A targeted diagnostics session against the affected user, device, and application can expose whether the bottleneck is endpoint resource usage, Web probe behavior, Cloud Path behavior, or a process-level condition.

> ⚠ **Warning**
> *Endpoint evidence must be time-bound*
>
> Do not conclude “hardware issue” simply because the device has some concerning metric somewhere in the history. Tie the endpoint signal to the affected time, affected application, and affected user experience. Otherwise, you are not doing RCA; you are doing laptop astrology.

Network Telemetry Analysis

Network telemetry in ZDX is designed to answer a practical question: where along the user-to-application path does degradation appear? The answer may be local Wi-Fi, first-hop routing, last-mile ISP, intermediate ISP, Service Edge path, private application path, or the final leg to the application. The administrator should read the network path as a set of legs rather than as one vague “internet” bucket.

Cloud Path analysis is the primary evidence source for path behavior. It can show direct paths, ZIA Public Service Edge paths, and ZPA paths where applicable. This matters because a user's route to an internet application and a private application may have different dependencies. The same endpoint can have a healthy direct path to one destination and a degraded private access path to another.

Cloud Path metrics include hop count, packet loss, and latency values such as average, minimum, maximum, and standard deviation. These metrics should be interpreted together. A stable path with moderate latency may be acceptable, while an unstable path with high variance or loss may create visible user impact even if the average latency does not look catastrophic.

Intermediate-hop loss requires caution. Routers and intermediate devices may rate-limit responses, so packet loss at one hop does not always indicate user-impacting loss. Loss at the destination, loss that continues across subsequent hops, or loss that aligns with poor application experience is more meaningful. In scenario terms, later-hop evidence usually carries more weight than a dramatic but isolated intermediate hop.

Incident-level views help move from individual troubleshooting to population troubleshooting. Device, Wi-Fi, Last Mile ISP, Intermediate ISP, ZIA Public Service Edge, ZPA, and Application incident categories each imply a different response path. If many users in a city or country are impacted through the same ISP or network segment, a single-user device workflow is the wrong path, even if one VIP happens to be the loudest ticket.

Network Intelligence-style reasoning also requires user impact validation. A network anomaly can be real without materially affecting user experience. If latency deviates from baseline but users do not show poor experience, the administrator should monitor and validate rather than over-escalate. Conversely, if anomaly windows overlap with poor user scores and application complaints, the evidence becomes operationally actionable.

A strong network conclusion compares affected and unaffected populations. Users on the same application but different ISPs, users on the same ISP but different applications, and users in the same location but different paths can all help isolate the affected leg. This comparative analysis is often the difference between an accurate escalation and a long meeting with five teams and no owner.

🎓 Exam Note

Cloud Path is evidence, not a verdict

Cloud Path helps identify where degradation appears, but it still requires interpretation. Intermediate loss, asymmetric paths, protocol behavior, and response rate limiting can affect what you see. Confirm the conclusion with user impact, destination behavior, affected population, and application telemetry before escalating.

Application Performance Indicators

Application performance indicators focus on whether the application is reachable, responsive, and returning the expected result. Web probes provide key metrics such as Page Fetch Time, DNS time, server response time, and availability. These indicators are especially useful when users report that a specific application is slow or unavailable.

Page Fetch Time is often the most visible application experience signal because it reflects how long the top-level page document takes to fetch. It does not represent every embedded object on the page, so the administrator should avoid overstating it as a full browser waterfall. Still, it is extremely useful for comparing performance across users, regions, and time windows.

DNS time and server response time help split application experience into useful subcomponents. If DNS time increases while the rest of the path is stable, name resolution becomes a stronger suspect. If server response time increases while DNS and network latency remain stable, application or hosting responsiveness becomes more likely. If Page Fetch Time increases while server response time remains normal, the administrator should examine the broader transaction and path context.

Availability depends on whether the probe receives an expected successful response. In many real deployments, a redirect to an authentication provider can be the correct response for an unauthenticated front-door probe. This makes response-code interpretation important. The administrator should know what “healthy” looks like for that application’s configured probe before treating every redirect or non-200 behavior as failure.

Application dashboards are useful for separating application-specific issues from shared-path issues. If multiple unrelated applications degrade for the same population, the root cause is more likely to be endpoint, Wi-Fi, ISP, or Service Edge path. If one application degrades while others remain healthy across the same users and locations, the application or its hosting path becomes more credible.

Collaboration and UCaaS applications add call quality dimensions such as latency, jitter, packet loss, and meeting participant impact. These signals should be evaluated separately from simple page access. A user may load the application successfully while media quality is poor because the media path, endpoint conditions, or relay selection is degraded. Collaboration troubleshooting is rarely one metric deep; if it were, meetings would have ended in 2014.

🔄 Cross-policy resources

ZDX with ZIA, ZPA, and Client Connector

ZDX does not replace ZIA, ZPA, or Client Connector troubleshooting. It guides where to look next. A private application symptom may require ZPA path and App Connector review. An internet or SaaS symptom may require ZIA path and Public Service Edge context. A device-only symptom may require Client Connector or endpoint operations review.

Identifying Hardware vs Network vs Application Issues

The cleanest way to separate hardware, network, and application issues is to use a hypothesis model. Start with the symptom, identify the affected scope, and then test the most likely domain using the telemetry that best proves or disproves it. The goal is not to inspect every possible metric. The goal is to eliminate domains until the evidence points to the most defensible root cause.

A hardware or endpoint hypothesis is strongest when the poor experience follows a specific device. Look for sustained resource pressure, repeated crashes or hangs, low device health, problematic process behavior, or a pattern tied to a device model, operating system, or software version. If the same user has a healthy experience on another device under similar network conditions, the endpoint hypothesis becomes much stronger.

A network hypothesis is strongest when the poor experience follows a path, location, ISP, SSID, or Service Edge route. Look for Cloud Path changes, packet loss that persists toward the destination, latency changes in a specific leg, Wi-Fi access point impact, last-mile ISP brownout or blackout indicators, or intermediate ISP anomalies. If unrelated applications degrade for the same population, application-specific blame becomes less likely.

An application hypothesis is strongest when the poor experience follows the application across multiple user populations. Look for high Page Fetch Time, increased server response time, availability failures, HTTP errors, unexpected redirect behavior, or an application-specific incident affecting multiple geographies. If users in different regions and on different ISPs show the same application-specific degradation while other applications are healthy, the application owner should be engaged.

Known-good comparison is one of the best ways to avoid guesswork. Compare the poor-score moment against a time when the user, device, application, and location were healthy. The meaningful differences may appear in Cloud Path metrics, Web probe metrics, endpoint telemetry, or device events. Automated analysis can accelerate this comparison, but the administrator still has to review whether the suggested factor matches the scenario.

Containment should be proportional to the evidence. A single device with high processor pressure does not justify a broad network escalation. A last-mile ISP issue affecting a country does not justify asking every user to reboot. An application availability issue does not justify changing Client Connector policy. Good containment narrows the blast radius; bad containment creates a new one.

Escalation should include the smallest complete evidence package. Include affected users or groups, affected locations, affected application, time window, score movement, Web probe indicators, Cloud Path findings, endpoint metrics, device events, and known-good comparison when available. A well-formed escalation saves the next team from rediscovering the same evidence. It also prevents the dreaded “please reproduce” loop, which is where tickets go to age poorly.

🎓 Exam Tip

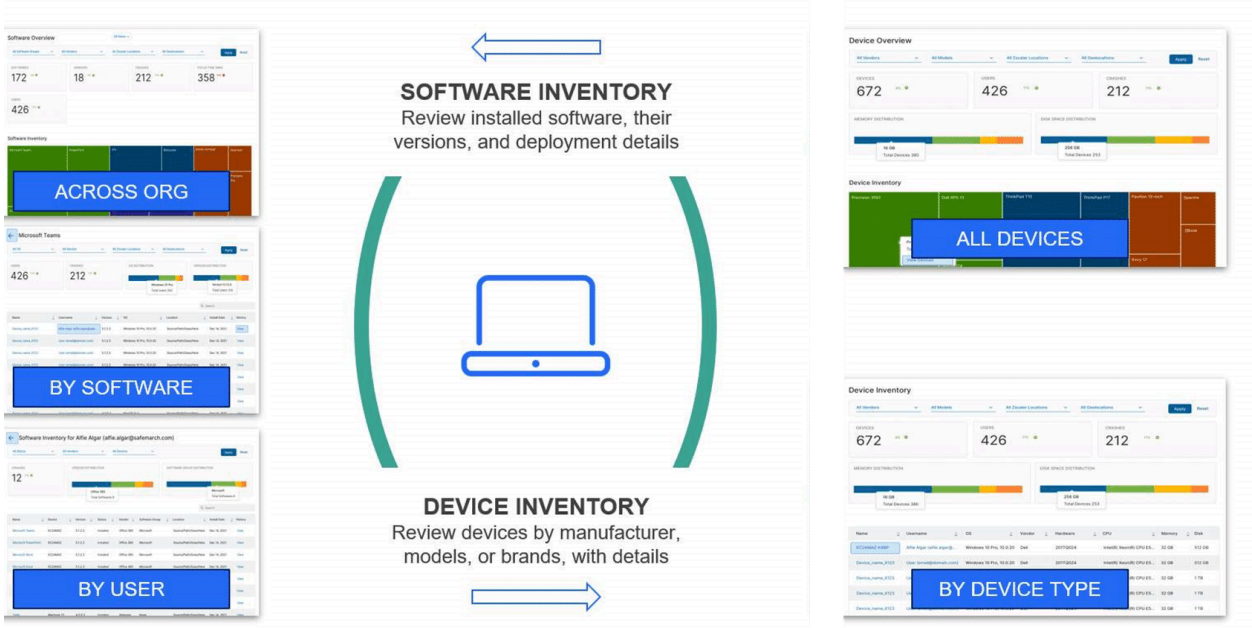
Match the owner to the evidence

- Endpoint evidence belongs with endpoint operations.
- SSID or access point evidence belongs with wireless operations.
- ISP or intermediate path evidence belongs with network operations.
- Server response time, HTTP errors, or application availability evidence belongs with the application owner.

Escalation quality is part of troubleshooting quality.

Device Inventory and Event Correlation

Device Inventory supports troubleshooting by showing which devices and users are in scope. Device name, user, operating system, vendor, hardware model, processor model, and installed memory can all matter when poor experience is concentrated around a specific hardware or operating system population. Inventory turns individual symptoms into fleet-level evidence.



Software Inventory adds version and installation context. When a problem affects only users running a particular browser, collaboration client, endpoint tool, or application version, software inventory helps define the impacted population. It also supports safer remediation because the administrator can identify who is actually affected before requesting software updates or endpoint changes.

Software lifecycle data can also matter. End-of-life or end-of-support software may not be the immediate root cause of a poor score, but it can explain instability, missing fixes, or repeated service desk patterns. In troubleshooting scenarios, this evidence should be used carefully: it

supports risk and remediation planning, but it should not be used as a shortcut around time-window correlation.

Device Events reports help correlate system crashes, blue screens, unexpected shutdowns, software crashes, and software hangs with user experience. The administrator should compare event timing against the affected application window. A crash before the score drop may explain the degradation. A crash after the user force-quits an application may be a consequence rather than the cause.

Process-level evidence helps identify whether a recurring process is contributing to poor device behavior. If the same process repeatedly appears across impacted devices during degraded experience windows, that supports a targeted remediation path. If only one user shows the process behavior, the response should stay narrow until additional evidence shows population impact.

The operational value of inventory and event correlation is proactive improvement. If a model, OS version, software version, or process pattern repeatedly maps to poor experience, the administrator can move beyond ticket-by-ticket troubleshooting. That is the ZDXA mindset: use today's RCA to prevent tomorrow's incident.

💬 Sidebar

Evidence quality for scenario reasoning

Strong scenario reasoning includes three elements:

- The observed signal: what changed and where.
- The correlation: which users, devices, locations, paths, or applications share the pattern.
- The action: what to validate, contain, remediate, or escalate next.
  Weak scenario reasoning usually skips correlation and turns one metric into a verdict.

📝 Quick Review

- Start every troubleshooting scenario by defining scope and time window.
- Use the ZDX Score to locate the affected experience, then review the underlying telemetry before deciding root cause.
- Endpoint indicators such as processor pressure, memory pressure, Wi-Fi signal, crashes, hangs, and restarts must align with the impacted time window.
- Cloud Path helps isolate where network degradation appears, including local, last-mile, intermediate, Service Edge, ZPA, and application-side legs.
- Intermediate-hop packet loss is not automatically user-impacting; destination loss and persistent loss across later hops carry more weight.
- Web probe metrics such as Page Fetch Time, DNS time, server response time, and availability help separate application behavior from path behavior.
- Application issues are more credible when one application is degraded across diverse users or regions while other applications remain healthy.

- Inventory and event correlation can reveal repeated patterns by device model, operating system, software version, process, or user population.
- Avoid common mistakes: blaming the application before scoping, treating one packet-loss hop as proof, ignoring device events, escalating without a time window, and confusing a low score with a completed RCA.

🏅 Exam Blueprint Alignment

Scenario Review

- Given a scenario, analyze device, network, and application telemetry to find performance issues across those tiers.
- Given endpoint telemetry, analyze processor pressure, memory utilization, Wi-Fi signal strength, software crashes, system crashes, and device inventory context to rule in or rule out hardware and local-device causes.
- Given network path information, identify bottlenecks across the end-to-end path, including local network, last-mile ISP, intermediate ISP, Public Service Edge, ZPA path, and application-side legs.
- Given ZDX score analysis or comparison output, use automated analysis and known-good comparison to accelerate root cause isolation.
- Given predefined or customized reports, analyze organization-level performance patterns rather than treating each ticket as isolated.
- Given organization-wide incident data, use ZDX incident categories such as device, Wi-Fi, last-mile ISP, intermediate ISP, ZIA Public Service Edge, ZPA, and application to prioritize response and escalation.
- Given device inventory, software inventory, process inventory, and device event evidence, correlate population-level patterns to support remediation planning and escalation.

Diagnostic Sessions and Log Analysis

Diagnostic sessions provide deeper evidence when ordinary dashboard telemetry is not enough to explain a user experience issue. They are most useful when the symptom is intermittent, user-specific, path-specific, process-related, or difficult to reproduce on demand. Instead of asking the user to become a part-time network engineer, Diagnostics collects targeted evidence from the user's actual device and path.

Log analysis supports the same workflow, but it answers different questions. Diagnostics can show what happened during the user's experience. Client Connector logs can help explain endpoint service behavior, forwarding state, network transitions, or local conflicts. ZIA logs can add policy and transaction context for internet and SaaS traffic. Audit logs help determine whether an administrative change occurred near the time of impact.

For this blueprint area, think in a sequence: initiate the right collection, interpret the output, review logs only where they add value, determine the probable root cause, remediate the smallest responsible component, and validate that the original experience improved. That

sequence matters because ZDXA scenarios are less about “where is the page?” and more about “what evidence justifies the next move?”

Initiating Diagnostic Sessions

Before starting a diagnostic session, verify that the environment can support it. The device must meet the minimum supported Zscaler Client Connector and ZDX module requirements, the subscription must support the selected diagnostic capability, and the administrator role must permit the action. Diagnostics also depends on active probe context on the device, so an administrator should not assume a session can be started against any arbitrary device at any arbitrary time.

A good session begins with a precise target: user, device, application, and time window. Starting from the User Overview or User Details context reduces selection mistakes because the relevant user and device details can be carried into the session. Copying an existing session can save time during repeated investigation, but copied values should still be reviewed. Yesterday’s correct target can become today’s beautifully preserved mistake.

Choose the diagnostic type based on the hypothesis. Deep Tracing is appropriate when the administrator needs granular user, device, application, web probe, Cloud Path, device health, and process evidence. Hi-Fi Cloud Path is better when the path itself is the focus and the administrator needs a higher-fidelity view of latency or packet loss. Bandwidth Testing helps evaluate connectivity and throughput characteristics. Capture probing should be reserved for packet-level questions.

Deep Tracing should usually be run for a short, intentional window. It can be configured to run from 5 minutes up to 60 minutes, and that range should be treated as an investigative control, not a default countdown timer. Run long enough to capture the symptom, but not so long that the collection becomes broad, stale, or operationally noisy.

When selecting the application and probes, match the collection to the service being investigated. For configured applications, select the relevant Web Probe, Cloud Path Probe, or both. A special application can be used when the issue involves a URL or tenant URL that is not already configured as a monitored application. For network applications that do not have a web front end, a single Cloud Path probe can be sufficient because Web Probe Metrics are not applicable.

Device Probing should remain enabled when endpoint performance is part of the question. Without it, the administrator loses visibility into device statistics, top processes, and user device events. That may be acceptable for a pure network-path investigation, but it weakens a user-experience investigation where CPU, memory, Wi-Fi, battery, disk, or network adapter behavior may be the actual cause.

🎓 Exam Tip

Start with a hypothesis

A diagnostic session should answer a specific question. “Is the user’s device saturated?” “Is packet loss persistent beyond the local gateway?” “Did DNS resolution spike at the same time as Page Fetch Time?” “Did the application stop returning an acceptable response?” If the collection does not map to a question, it is probably too broad.

⚠️ Warning

Do not over-collect

Capture probing and broad diagnostics can expose more data than the investigation requires. Use specific users, devices, applications, filters, interfaces, and time windows. Least privilege applies to troubleshooting data too; packet captures are not confetti.

Interpreting Diagnostic Output

Begin with the session metadata before interpreting graphs. The session name, type, user, device, created time, start time, status, application, and creator establish the evidence boundary. Created Time indicates when the administrator created the session. Start Time indicates when Zscaler Client Connector accepted the request and began collection. Mixing those two timestamps can shift the analysis window and lead to the wrong conclusion.

Status determines how much trust to place in the results. Created means the session was created but has not yet started. Started means Client Connector acknowledged the request and probe execution is expected to begin. In Progress means the session is actively executing and updating data. Completed indicates a successful collection. Expired means no usable data is available because the device did not respond in time. Failed, Incomplete, Aborted, and Abort Initiated should be treated as partial or interrupted evidence.

In Progress sessions can update data every minute, which is useful during an active troubleshooting bridge. Completed sessions show full session data. Other historical states may show partial data, depending on where the collection failed or stopped. The administrator should say “the evidence is incomplete” when it is incomplete, not build a castle out of half a packet and a good attitude.

The strongest diagnostic interpretation is time-aligned. Select the same point in time across Web Probe Metrics, Cloud Path, Device Health, Top Processes, and User Device Events. A CPU spike that occurred 20 minutes after the user’s score drop is not proof of endpoint root cause. A DNS spike, path loss, and device event at the same timestamp is a very different story.

Web Probe Metrics explain the application transaction. Page Fetch Time measures the time required to fetch the top-level page document from the configured URL. DNS Resolve Time shows name-resolution delay. Server Response Time helps isolate how long the server took to provide the first byte of response. Availability indicates whether the probe received an acceptable response. These metrics should be interpreted together, not as isolated trivia.

Cloud Path explains the route and performance between the user and the destination. The graph can show latency or packet loss over time. Hop View and Command Line View show the path details. Direct paths, Internet and SaaS Service Edge paths, and Private Access Service Edge paths have different legs, so the administrator should identify the path type before deciding where the bottleneck belongs.

Packet loss in intermediate hops requires caution. Some routers rate-limit or deprioritize diagnostic traffic, so intermediate loss does not always mean user traffic is failing. Destination behavior and persistence across subsequent hops matter more. If later hops and the destination do not show loss, do not build a root cause around a dramatic-looking middle hop. Traceroute has been lying to network engineers since before some candidates had Wi-Fi.

Device Health and Top Processes help determine whether the endpoint could deliver a good experience at the time of the issue. CPU, memory, disk, network, Wi-Fi, and battery metrics can reveal local constraints. Top Processes identify which processes consumed memory, CPU, disk I/O, or network I/O. User Device Events can show restarts, sleep events, network changes, Wi-Fi transitions, or adapter changes that correlate with the experience drop.

Automated Root Cause Analysis can accelerate interpretation by identifying probable factors, explanations, and confidence levels for a poor score. It is most useful when paired with Compare, because comparing an affected point to a known-good point highlights what changed. Treat the output as guided analysis. It can reduce time to resolution, but the administrator still validates the conclusion against the underlying telemetry.

🎓 Exam Note

Correlation is the administrator's job

ZDX exposes the signals, but the candidate must connect them. The likely root cause is the signal that explains the score drop, aligns with the user's reported time, persists at the responsible layer, and is not contradicted by adjacent metrics.

Reviewing Log Files

Log review should be selective. The diagnostic report is usually the first artifact for a ZDX-centered experience issue because it preserves the relevant web, path, device, process, and event data. Additional logs should be collected only when they answer a question that the diagnostic result cannot answer.

Time alignment is the first rule of log analysis. Align the user report, ZDX Score change, diagnostic Created Time, diagnostic Start Time, diagnostic End Time, Client Connector events, and any related transaction logs. If the logs use different time zones or if the user describes the issue in local time, normalize the timeline before interpreting the evidence.

Client Connector logs are most useful when the suspected cause involves endpoint behavior. Examples include local service state, forwarding behavior, tunnel state, device network changes, driver or adapter behavior, captive portal conditions, local conflicts, or issues that appear only on a specific device. Debug-level collection should be used deliberately and only when normal evidence is insufficient.

ZIA transaction logs can provide supporting evidence when the affected traffic traverses the Internet and SaaS path. They can help validate whether the expected request occurred, whether policy or inspection affected the transaction, and whether Deep Tracing probe activity is visible during a live troubleshooting window. For private application scenarios, ZPA path context, Service Edge selection, App Connector reachability, and application-side evidence may be needed to complete the story.

Packet capture evidence should be reviewed only against a specific packet-level hypothesis. A scoped capture can validate DNS behavior, TCP connection setup, TLS negotiation, retransmissions, resets, or whether traffic is leaving the expected interface. If the capture is not scoped by destination, port, interface, frame size, and storage limits, the investigation can quickly turn into a haystack factory.

Audit logs matter when the issue might be configuration-induced. If a monitored application, probe, alert rule, role, webhook, integration, or administrator setting changed just before the issue began, the audit trail can turn a mystery into a change-control conversation. That is not glamorous, but neither is chasing packet loss for three hours when somebody edited the probe URL at lunch.

🔄 Cross-policy resources

ZDX, ZIA, ZPA, Client Connector, and platform logs

Use ZDX as the experience source of truth, then bring in adjacent evidence only when needed. ZIA helps with internet and SaaS transaction context. ZPA helps with private application path context. Client Connector logs help with endpoint-side service and forwarding behavior. Audit logs help identify administrative changes.

Determining Root Cause

Determining root cause starts by classifying the impact pattern. If the issue affects one user on one device, start with endpoint health, local network, Wi-Fi, and that user's path. If the issue affects many users at one location, evaluate Wi-Fi, gateway, ISP, and site egress behavior. If the issue affects many users across regions for one application, evaluate application availability, server response, SaaS service behavior, and shared provider paths.

For single-user cases, the administrator should look for evidence that is unique to that user or device. Device Health, Top Processes, User Device Events, Wi-Fi signal, adapter changes, and local Client Connector behavior often matter more than organization-wide application dashboards. A single user with poor CPU, weak Wi-Fi, and normal application metrics is not an application outage. It is a laptop trying to win a losing footrace.

For site, city, or ISP cases, the administrator should look for shared-path evidence. Cloud Path latency, packet loss, last-mile ISP behavior, Wi-Fi access point patterns, and incident grouping can reveal whether many users are affected by the same infrastructure segment. The key is to prove the common element. "Everyone is slow" is less useful than "users on this ISP in this city show the same leg latency increase during the same interval."

Application root cause becomes more likely when endpoint and path metrics are stable but Web Probe Metrics degrade. High Server Response Time, failed availability, unexpected response codes, redirect changes, or sustained Page Fetch Time increases for multiple users can point toward the application or its front end. Before escalating to the application owner, verify that the probe definition still reflects the application's expected behavior.

Private application scenarios require path segmentation. A user-to-egress problem, a Service Edge leg issue, an App Connector leg issue, and an application-side issue all produce different operational responses. ZDX can help visualize the path, but the administrator must decide which leg owns the evidence. This is where “private app is slow” becomes a structured path analysis instead of a group chat with 14 people and zero conclusions.

Device and software root cause should be supported by both symptoms and telemetry. High CPU, memory pressure, disk I/O, network I/O, weak Wi-Fi, software crashes, outdated software, or a noisy process can explain poor experience when the timing lines up. Inventory and process data can also show whether the issue is isolated or tied to a software version, hardware model, operating system, or common process.

Automated Root Cause Analysis can identify likely factors and confidence levels, but candidates should treat it as evidence acceleration rather than evidence replacement. The Compare workflow is especially useful because it shows differences between an affected point and a known-good point. The most defensible conclusion is one that matches the automated factor, the graphs, the events, and the user’s reported symptom.

A clean root cause statement should include the affected population, affected application or path, time window, causal layer, supporting metrics, and validation plan. For example, “Users in a specific city experienced elevated Page Fetch Time because Server Response Time increased for the application during the same interval” is stronger than “the app was slow.” Root cause statements should be boringly precise. Boringly precise wins escalations💬 Sidebar

💬 Sidebar

Root cause versus symptom

A symptom is what the user experienced. A signal is what ZDX measured. A root cause is the explanation that connects the symptom and signal to a responsible layer. Do not stop at “poor score,” “packet loss,” or “high CPU.” Explain why that signal caused the experience impact and why competing causes were ruled out.

Implementing and Validating Remediation

Remediation should be the smallest safe action that addresses the proven cause. If one device is affected, do not change a tenant-wide setting. If one access point is weak, do not rewrite the application monitoring strategy. If one probe is misconfigured, do not declare an application outage. Good administration is controlled change with measurable validation.

For network-related remediation, the action depends on the affected segment. Wi-Fi issues may require moving the user, changing access points, addressing weak signal, reducing retransmissions, or escalating to the WLAN team. Last-mile or intermediate ISP issues may require provider escalation with Cloud Path evidence. Forwarding, MTU, routing, or tunnel-related issues may require Client Connector or network configuration review before broader change.

For application-related remediation, separate monitoring configuration from application behavior. If the probe URL, response code list, redirect handling, timeout, or expected authentication flow no longer matches the application, fix the monitoring definition. If the probe is correct and multiple users show degraded application metrics, escalate with affected regions, time range, Page Fetch Time, Server Response Time, availability, and response behavior.

For endpoint-related remediation, focus on the observed constraint. Close or update a high-impact process, correct a Wi-Fi problem, address memory or CPU pressure, patch or remove problematic software, remediate a crash pattern, or guide the user through a supported Client Connector recovery step. Restarting services, clearing ZDX data, or collecting new logs should be reserved for cases where endpoint collection or local service state is part of the hypothesis.

Containment is different from remediation. Containment limits user impact while the root cause is being fixed. That might mean routing a ticket to the correct team, advising affected users, muting a noisy but understood alert, temporarily adjusting a narrowly scoped probe, or isolating a bad software version. Containment should not hide the problem so well that nobody fixes it. That is not containment; that is putting a blanket over the dashboard.

Automation and integrations can reduce time to resolution, but they must be governed. API access should use appropriately scoped credentials, protected key material, and the minimum permissions required. ITSM workflows can initiate Deep Tracing, run root cause analysis, create incidents, or route alerts, but automated actions should still preserve user scope, device scope, and approval boundaries. Least privilege is not just for security policy; it is for operations too.

Validation must use the same evidence family that proved the issue. If Page Fetch Time was the problem, validate Page Fetch Time recovery. If Cloud Path latency was the problem, validate the affected leg. If a process caused CPU saturation, validate CPU, Top Processes, and user experience after remediation. If a configuration was changed, verify that the expected metric improved and that no adjacent population was harmed.

Finally, document the result in a form another operations team can trust. Export diagnostic results or share a snapshot when appropriate, and obfuscate user or device details when privacy requirements apply. The best closure note states what changed, what evidence proved the cause, what action was taken, what metric recovered, and what monitoring remains in place. Future-you will appreciate present-you. Future-you is notoriously picky.

📝 Quick Review

- Start with scope before cause: user, device, site, city, ISP, Service Edge path, application, or organization.
- Use ZDX Score as a triage signal, not as the root cause itself.
- Initiate Diagnostics only after defining the user, device, application, time window, and hypothesis.
- Use Deep Tracing for granular user, device, application, process, web, and Cloud

Path evidence.

- Use Hi-Fi Cloud Path when path fidelity matters; use capture probing only for packet-level questions.
- Interpret session status before trusting results; Completed, In Progress, Expired, Failed, Incomplete, and Aborted do not carry the same evidentiary weight.
- Correlate Web Probe Metrics, Cloud Path, Device Health, Top Processes, and User Device Events at the same timestamp.
- Treat intermediate hop loss carefully; destination behavior and persistence across later hops matter more.
- Review logs selectively: diagnostic reports, Client Connector logs, ZIA transaction logs, ZPA path context, audit logs, and packet captures each answer different questions.
- Validate remediation with the same metric family that identified the issue.
- Common mistakes to avoid: changing global settings for a local issue, blaming the application before validating the probe, ignoring Created Time versus Start Time, collecting broad packet captures, and accepting automated analysis without checking the supporting telemetry.

🏅 Exam Blueprint Alignment

Scenario Review

- Analyze device, network, and application telemetry to identify performance issues across those tiers.
- Initiate diagnostic collection for a user, device, application, or path based on the reported scenario.
- Interpret diagnostic output from Deep Tracing, Hi-Fi Cloud Path, Bandwidth Testing, and capture probing.
- Review diagnostic artifacts and supporting logs to correlate user reports, session timing, Client Connector behavior, and traffic evidence.
- Analyze Cloud Path information to locate bottlenecks across local network, ISP, Internet and SaaS Service Edge, Private Access Service Edge, App Connector, and destination application segments.
- Use automated root cause analysis and comparison workflows to accelerate investigation while validating the recommendation against underlying metrics.
- Leverage incidents and dashboard context to distinguish isolated user issues from Wi-Fi, last-mile ISP, ZIA, ZPA, device, or application incidents.
- Share current-state evidence through exported diagnostics or snapshots when escalation requires another operations team to continue the investigation.
- Implement remediation that is scoped to the proven failure domain and validate the outcome using the original metric evidence.

Network Path Analysis and Bottleneck Identification

Network path analysis is where ZDX moves from user sentiment to path evidence. Cloud Path shows how traffic moves from the user device toward the application or destination, and it provides latency, packet loss, hop, protocol, tunnel, and Service Edge context where available. This lets an administrator identify whether a bottleneck appears near the user, at local egress, across the ISP, through a Service Edge segment, or closer to the application.

This subsection maps most directly to the troubleshooting objective that asks candidates to analyze network path information to find bottlenecks through the entire path. It also supports broader telemetry analysis across device, network, and application tiers. In exam scenarios, expect the path data to be only part of the evidence. The better answer often requires correlating Cloud Path with Web Probe metrics, device health, incident scope, and current-versus-baseline behavior.

The goal is not to prove that a network hop has a number attached to it. The goal is to decide what action is justified. A user-local issue may require endpoint or Wi-Fi remediation. A Last Mile ISP or Intermediate ISP issue may require provider escalation or temporary routing strategy. An application-side delay may require application owner escalation. A Service Edge path issue requires preserved Cloud Path context and disciplined evidence collection.

End-to-End Path Visibility

End-to-end path visibility in ZDX is designed to represent the user’s actual access experience, not just application uptime from a monitoring point in a data center. Cloud Path provides path visualization between hop points, and it can represent a direct path from Zscaler Client Connector to egress and then to the destination, or a tunneled path through an Internet & SaaS Public Service Edge.

Cloud Path does not stand alone. Web Probe metrics explain whether the monitored application transaction is responding properly, while Cloud Path explains the network path conditions around that transaction. When both are reviewed for the same application and time window, the administrator can separate “the path is slow” from “the application is slow” with far more confidence.

Hop View and Command Line View serve different troubleshooting purposes. Hop View is the visual orientation layer: it quickly shows the user device, egress, tunnel or Service Edge segment, and destination path. Command Line View is the more MTR-style layer: it is better suited for reviewing hop data, packet loss, failed packets, latency, and path details in a structured format.

Tunnel information must be interpreted as part of the path, not as decoration. In the Cloud Path section, ZDX can display GRE tunnel, IPSec tunnel, Z-Tunnel 1.0, Z-Tunnel 1.0 over GRE, Z-Tunnel 1.0 over IPSec, Z-Tunnel 2.0, Z-Tunnel 2.0 over GRE, and Z-Tunnel 2.0 over IPSec. If

the client uses a Direct Request to reach the application, the traffic bypasses the tunnel and no tunnel graphic appears in Hop View.

The path can also include Service Edge context. For internet applications, the path can show traffic from the client toward the application with an Internet & SaaS Public Service Edge in the path when applicable. For private applications, the path can show the user side of the Private Access path. In some designs, both Internet & SaaS and Private Access service components can appear, which is a clue that the administrator must reason about both the internet forwarding path and the private application path.

Visibility remains conditional. A tunnel, external proxy, protocol block, No Default Route environment, or unreachable egress device can limit what ZDX can observe. When Cloud Path reports that client egress cannot be traced, protocol detection failed, or hop data is not discoverable, the administrator should treat that as a visibility constraint and adjust the investigation rather than forcing a conclusion that the data does not support.

Device context matters in end-to-end analysis. The network interface type, device location method, private and public IP information, operating system, Client Connector version, and device health can all change how the path should be interpreted. A road-warrior device on cellular, a laptop behind a corporate GRE tunnel, and a workstation on office Wi-Fi can produce very different path evidence for the same application.

End-to-end path visibility also helps contain blast radius. If one user has a poor experience while other users in the same region and application are healthy, the investigation should move toward endpoint, local network, or user-specific path. If many users in a region or ISP share the same path degradation, the administrator should move toward incident, ISP, or routing analysis. The path tells the story; the population tells whether it is a short story or a novel.

🎓 Exam Tip

Orient, then verify

Use Hop View to quickly understand the traffic path. Use Command Line View, Web Probe metrics, device health, and incident scope to verify the conclusion before choosing remediation or escalation.

Identifying Latency Sources

Identifying latency sources begins with time alignment. The Cloud Path latency graph can show latency over a selected period, and selecting a point updates the path view for that moment. This matters because latency from the wrong time window is noise, not evidence. Always align the path view with the user complaint, ZDX Score drop, incident window, or diagnostics session.

A high-latency leg is a lead, not a verdict. Hop View can highlight the highest-latency leg, and differential latency helps show where the largest increase appears. That is useful for narrowing

the investigation, but it must be checked against user impact, later-hop behavior, Web Probe data, and baseline or comparison information before it becomes root cause.

Web Probe metrics provide the application transaction lens. Page Fetch Time reflects how long the selected application takes to transfer or load the monitored page. DNS Time, TCP connection time, SSL handshake time, server response or processing time, redirects, and availability help break that transaction into parts. If the path looks healthy but Page Fetch Time or server response degrades, the application side deserves more attention.

A clean scenario pattern is to compare path health with application health. If Cloud Path latency or packet loss rises while Web Probe metrics remain otherwise normal, the network path is more likely to be involved. If Page Fetch Time, server response, HTTP errors, or availability degrade while Cloud Path remains stable, the application path or application service is more likely. When both degrade, the administrator should look for shared dependencies such as DNS, egress, ISP, Service Edge segment, or application-side network reachability.

Baseline and comparison views are especially valuable because “normal” differs by geography, ISP, application, and user population. A candidate should be comfortable comparing a poor point to a last known good score or another relevant time. The purpose is to identify which metric materially changed, not just to prove that the score changed.

Network Intelligence, when available, expands latency reasoning from individual Cloud Path evidence to network-wide anomaly analysis. It uses Cloud Path metrics such as latency and packet loss to establish baselines and identify Last Mile or Intermediate ISP anomalies. The important decision criterion is whether the anomaly correlates with poor user experience, not merely whether the anomaly is statistically visible.

Diagnostics sessions are appropriate when normal dashboard evidence is not granular enough. During a diagnostics session, ZDX can collect Web Probe, Cloud Path, and device statistics at a tighter interval, along with device health, top process, and user device event information when enabled. This is useful for intermittent or user-specific issues where the administrator needs a higher-resolution record.

Automated root cause analysis can accelerate the investigation by identifying likely contributors to a poor score and comparing conditions across time. It should be treated as an analysis aid, not a substitute for judgment. The administrator should still verify that the suggested factor aligns with Web Probe metrics, Cloud Path data, device health, and the affected population.

Latency decisions should translate into ownership. A first-hop or Wi-Fi-related pattern points toward local network or endpoint action. An ISP or ASN pattern points toward provider or routing investigation. A Service Edge-adjacent pattern requires preserving Service Edge and time-range details. An application-side delay points toward application owner evidence. The exam favors the candidate who can explain why the next action follows the data.

> ⚠️ **Warning**
> *Latency is not always user impact*
>
> A latency anomaly can be real without materially affecting users. Confirm whether poor ZDX Scores, Web Probe degradation, incident impact, or user complaints line up with the latency window before declaring root cause.

Local Network vs ISP vs Application Bottlenecks

A local-network bottleneck usually appears close to the user. Evidence can include Wi-Fi signal problems, access point latency, first-hop packet loss, frequent network changes, weak interface conditions, or degradation limited to one user, one device, one SSID, or one site. If the same application is healthy for nearby users on different networks, the local hypothesis becomes stronger.

Device bottlenecks should not be skipped just because the symptom sounds like a network issue. User Details and device health views can show CPU, memory, disk, network, battery, Wi-Fi, and device event information. High CPU or memory pressure, a restart during the complaint window, unstable network activity, or a resource-heavy process can explain a poor application experience even when the network path is clean.

Shared Wi-Fi evidence changes the response. If multiple users on the same SSID or access point show poor scores, access point latency, packet loss around early hops, or weak signal, the likely ownership shifts from individual endpoint remediation to wireless infrastructure review. This distinction matters in scenario questions because treating every user as a separate endpoint case wastes time and hides the real incident.

A Last Mile ISP bottleneck is more likely when users in the same geography or ISP show shared degradation. Brownout-style patterns can show performance degradation, while blackout-style patterns can show connectivity failures. Useful evidence can include ZDX Score drop, DNS latency, leg latency, packet loss, application score, probe errors, and Cloud Path probe errors depending on the incident subtype.

Intermediate ISP issues sit beyond the user's immediate access provider and can involve ISP backbone or peering paths. Network Intelligence can help identify affected ASNs, baseline deviation, latency, packet loss, impacted applications, and whether users actually have poor digital experience. Candidates should distinguish last-mile ownership from intermediate-path ownership because the escalation path and containment options differ.

Application bottlenecks become more likely when users across different local networks or regions show the same application degradation while other applications remain healthy. Web Probe evidence is central here. HTTP errors, high server response time, an unfavorable TTFB-to-PFT relationship, excessive redirects, application availability failures, or high last-server-leg latency can all support an application-side conclusion.

Tunnel and forwarding design can create or obscure bottlenecks. Z-Tunnel over GRE or IPSec, location tunnels, external proxies, and direct requests change the observable path and the owner of the next action. If the Cloud Path cannot trace client egress because the path is tunneled, the administrator must avoid claiming full end-to-end latency visibility and should document the limitation.

Private application scenarios require additional care. A Private Access-related path can involve policy, attributes, app segment configuration, App Connector reachability, and Service Edge behavior. If the evidence indicates a policy or attribute mismatch, an App Connector group issue, or an application not configured condition, the administrator should not mislabel it as generic packet loss. Private access failures are still user experience issues, but their cause may be policy or application access control rather than raw network performance.

The operational action should match the evidence. Local issues call for endpoint, Wi-Fi, or LAN remediation. ISP issues call for provider evidence and possible routing or workaround discussions. Application issues call for application-owner escalation with Web Probe and affected-user evidence. Service Edge or visibility issues call for preserved Cloud Path, tunnel, protocol, and timestamp details. In all cases, containment starts by narrowing the affected population before changing anything broad.

🔄 Cross-policy resources

Where ZDX meets access policy

ZDX can show symptoms that originate outside ZDX itself. Private Access policy, SAML or SCIM attributes, app segment state, Client Connector forwarding profile, tunnel configuration, firewall rules, and proxy behavior can all influence the user experience evidence. Keep ZDX central, but do not ignore the control plane that shaped the path.

Interpreting MTR-Style Path Data

MTR-style interpretation is the discipline of reading hop-by-hop data without overreacting to every intermediate response. Command Line View is the closest ZDX view for this style of analysis because it provides a structured path record that can be reviewed more like traditional traceroute or MTR output. It is useful for identifying where latency or loss first appears and whether it continues downstream.

The safest packet-loss rule is persistence. Loss on one intermediate hop may be caused by rate limiting or probe-response behavior on that device. Loss that begins at a hop and continues through later hops or reaches the destination is stronger evidence of an actual path impairment. Destination behavior matters more than a single dramatic intermediate number.

Latency should be interpreted both cumulatively and differentially. Cumulative latency tells whether the overall path is expensive. Differential latency tells where the major increase occurs. A high first leg points toward client, Wi-Fi, gateway, or local egress. A later increase around ISP

or peering segments points toward provider path. A final-segment increase should be compared with Web Probe and application metrics before concluding the destination is at fault.

Protocol choice matters because path visibility depends on what the network permits. Cloud Path can use ICMP, TCP, UDP, and in supported cases Adaptive mode. Adaptive mode can select the best available protocol per leg, but it is not supported for internal applications through Private Access; for those cases, ICMP, TCP, and UDP are the supported protocol choices. In No Default Route environments, ICMP and UDP probes are not supported, so TCP should be used.

Cloud Path errors should be interpreted as part of the MTR-style evidence set. A network communication failure can indicate that an interface is down, traffic is blocked, or upstream connectivity is unavailable. A domain resolution error points toward the configured name or DNS. A client egress detection problem can indicate that the configured Cloud Path protocol is blocked. These messages often tell the administrator whether to change the probe protocol, validate DNS, check firewall behavior, or preserve evidence for Support.

Command Line View is also useful for escalation because it provides a compact technical record. The administrator should capture the time range, affected user or population, application, packet loss pattern, latency pattern, protocol, tunnel type, and Service Edge context. Where a Service Edge identifier appears, preserve it because it can help correlate the path to the specific service component involved.

When a tunnel or proxy is present, MTR-style conclusions need extra caution. GRE, IPSec, external proxy, VPN concentrator, and Z-Tunnel combinations can affect both the real forwarding path and the observable probe path. If the path data excludes the client-to-egress segment or reports that a section is not discoverable, the analysis should state that limitation instead of pretending the path is fully visible.

The final skill is concise escalation. A high-quality escalation does not say, “The network is slow.” It says which users are affected, which application is affected, when the issue occurred, where loss or latency begins, whether it persists to later hops or destination, whether Web Probe metrics agree, what tunnel and protocol were used, and what local or application evidence was ruled out. That level of precision is how ZDX reduces mean time to resolution instead of just producing prettier tickets.

📝 Quick Review

- Start with scope: user, device, SSID, site, ISP, region, application, or widespread incident.
- Use ZDX Score to find where to investigate; use underlying metrics to determine why.
- Align all evidence to the same time window before interpreting path, Web Probe, or device data.
- Use Hop View for orientation and Command Line View for MTR-style validation.
- Read tunnel context carefully: GRE, IPSec, Z-Tunnel versions, Z-Tunnel over GRE or IPSec, and Direct Request change path interpretation.

- Treat no tunnel graphic during Direct Request as expected behavior, not automatically as missing data.
- Isolated intermediate-hop loss is often less important than loss that persists to later hops or the destination.
- Use Web Probe metrics to separate DNS, TCP, TLS, server response, redirects, availability, and Page Fetch Time contributors.
- Use Network Intelligence to evaluate Last Mile and Intermediate ISP patterns when the issue appears broader than one user.
- Use diagnostics sessions when normal dashboard evidence is not granular enough for an intermittent or user-specific issue.
- Avoid common traps: blaming the largest latency number without impact evidence, ignoring device health, ignoring tunnel visibility limits, and escalating without timestamps.

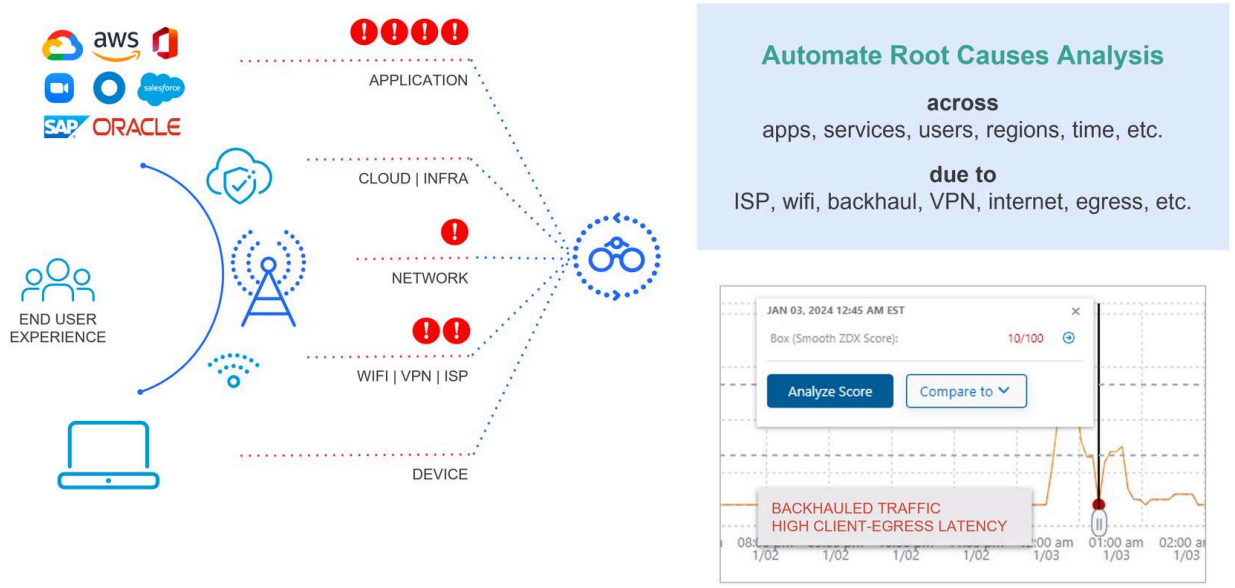
🏅 Exam Blueprint Alignment

Scenario Review

- Section 3.1: Analyze device, network, and application telemetry to determine which tier is causing performance degradation.
- Section 3.3: Analyze network path information to identify bottlenecks across the full path, including local network, ISP, intermediate path, Service Edge, and application-side segments.
- Section 3.4: Use automated analysis and score comparison to accelerate root cause analysis while validating the recommendation against telemetry.
- Section 1.2: Interpret dashboard scope to distinguish single-user, site, regional, application-specific, and widespread incidents.
- Section 1.8: Analyze Last Mile or Intermediate ISP performance and determine whether network anomalies are materially affecting users.

Using Zscaler AI for Root Cause Analysis

Zscaler AI capabilities in ZDX help administrators reduce the time spent manually comparing telemetry across dashboards, diagnostics, incidents, and user details. These capabilities can identify likely contributing factors, summarize operational data, and expose patterns that might otherwise require a senior engineer to piece together by hand. The goal is not to replace troubleshooting discipline; the goal is to make the first hypothesis better and faster.



The blueprint maps this area to scenario-based analysis: given user complaints, score degradation, diagnostic output, or a path bottleneck, candidates must decide how to use AI-assisted insight without skipping validation. AI can highlight a probable factor such as DNS delay, path latency, Wi-Fi degradation, device resource pressure, or application response behavior. The administrator still needs to confirm the factor using the underlying telemetry before remediation or escalation.

The most reliable pattern is to treat AI output as a guided hypothesis. If AI identifies network latency, inspect Cloud Path latency, packet loss, and impacted path legs. If AI identifies endpoint health, inspect Device Health, Top Processes, Process Inventory, and User Device Events. If AI identifies application behavior, inspect Page Fetch Time, Server Response Time, DNS Resolve Time, Availability, and whether the issue is isolated to one application or shared across several.

🎓 Exam Note

AI output is evidence-adjacent, not evidence

A ZDX AI insight should guide the investigation. It becomes operational evidence only after it is confirmed with the relevant ZDX telemetry.

Copilot for Troubleshooting

ZDX Copilot is an AI-powered conversational assistant that complements ZDX by helping administrators retrieve data, summarize user or incident context, recognize rising trends, and produce relevant visualizations. For troubleshooting, its value is strongest when the administrator asks precise questions that include a user, application, time range, incident type, or metric. A vague prompt produces vague value; the help desk already has enough fog machines.

Copilot can help accelerate user-level investigations. For example, an administrator can ask about a user's ZDX Score for a specific application over the last few hours, page fetch time for a SaaS application, Wi-Fi signal quality, or Microsoft Teams call quality behavior. The output should then be cross-checked in the User Details page, web probe graphs, Cloud Path view, device health graphs, and any available call-quality context.

Copilot can also help with incident-level triage. When several tickets arrive close together, an administrator can ask Copilot to summarize incidents in a recent time window, identify ongoing alerts, or show applications impacted by alerts. This is useful for separating an emerging shared incident from unrelated single-user complaints. The next decision is whether the pattern belongs to Wi-Fi, Last Mile ISP, Service Edge path, ZPA path, application, or endpoint operations.

When using Copilot for troubleshooting, candidates should think in terms of prompt design. A strong prompt narrows scope: "Analyze the impacted users with poor ZDX Scores for SharePoint in the last four hours and identify whether the common factor is device, network, or application." A weak prompt asks for a conclusion without supplying enough operating context. Copilot is a good analyst; it is not a mind reader, even on a good caffeine day.

Access control matters. ZDX Copilot requires the appropriate subscription level and permissions. In an operational design, this means Copilot access should be assigned intentionally to roles that need interactive troubleshooting assistance. A service desk role may need enough access to summarize user experience and incidents, while broader administrative or configuration guidance should remain aligned to least-privilege role design.

Copilot can provide configuration guidance for probes, alerts, administrator roles, and webhook-style operational integrations, but configuration guidance should not be confused with approved change execution. The administrator still needs to confirm the object type, criteria,

thresholds, affected users, and notification path. This is especially important when a suggested alert or workflow could create noise, duplicate tickets, or route an incident to the wrong operations queue.

ZDX Copilot retains prompts and responses for a rolling session history period. Administrators should be mindful of what they enter and should avoid using prompts as a substitute for ticket notes or approved incident documentation. The durable record should contain the validated conclusion, supporting telemetry, scope, and corrective action, not just the conversational transcript.

> ⚠️ **Warning**
> *Do not turn Copilot into a magic 8-ball*
>
> Copilot can summarize, correlate, and recommend. It should not be used to justify remediation unless the recommendation is validated against the same ZDX telemetry an administrator would use manually.

Deep Trace Analysis with AI

Deep Trace analysis belongs in the workflow when the standard dashboards identify a problem but do not provide enough resolution to explain it. In current ZDX operations, Deep Tracing is used as a Diagnostics session type that can gather higher-granularity evidence for a selected user, device, and application. It is especially useful when the issue is intermittent, user-specific, process-related, or difficult to reproduce while the support team is watching.

A Diagnostics session can collect Web probe information, Cloud Path information, and device statistics during the session. The administrator can use the results to correlate application fetch behavior, network path behavior, device health, top process consumption, and user device events in the same time window. This time alignment is the value. Without it, teams often compare a user complaint from 9:05 with network data from 9:40 and then wonder why the story reads like bad fiction.

Before starting a Diagnostics session, the administrator must satisfy operational prerequisites. The endpoint needs supported versions of Zscaler Client Connector and the ZDX Module, the subscription must support Diagnostics, and the admin role must have the required Diagnostics permission. These requirements matter in exam scenarios because the correct answer may be to verify capability or permission before attempting to run the session.

Diagnostics should be targeted. The administrator selects a session type such as Deep Tracing, Hi-Fi Cloud Path, Bandwidth Test, or Packet Capture Probing based on the suspected failure domain. Deep Tracing is appropriate when the issue may involve process-level endpoint behavior, web probe metrics, device state, and path data. Hi-Fi Cloud Path is appropriate when stronger path evidence is needed. Packet Capture Probing is reserved for cases where packet-level evidence is justified.

A Diagnostics session also depends on probe context. An active probe is required on the device, and the probe must have run long enough to provide usable context before the session starts. If the application is already monitored, the administrator can select the relevant Web probe or Cloud Path probe. If the issue involves a destination that is not already configured as a monitored application, a special application URL can be used for the session without redesigning the ZDX application catalog.

Deep Tracing can expose process-level evidence that is not obvious from a standard score view. If the user reports slowness and the session shows stable path latency and normal application response but a CPU spike tied to a specific process, the likely investigation path moves toward endpoint operations. If device health is stable but Cloud Path latency spikes on a specific leg, the likely investigation path moves toward network or ISP evidence. The administrator should let the session data move the case, not their favorite theory.

Session status also has troubleshooting value. In-progress sessions update while the session is running; completed sessions provide complete data; incomplete or failed sessions may provide partial data; expired sessions may indicate that the device was offline or did not respond in time. Candidates should not treat missing session data as proof that no issue exists. Sometimes the diagnostic failure is itself a clue about endpoint reachability or network state.

Exported Diagnostics results are useful when escalation is required. The artifact should preserve the user, device, application, time window, selected probes, key Web metrics, relevant Cloud Path section, device health indicators, process-level evidence, and any notable device events. A good escalation package lets the next team act without reopening the entire investigation from scratch.

🔄 Cross-policy resources

ZIA, ZPA, and Client Connector context

Deep Trace evidence may intersect with ZIA, ZPA, Client Connector, endpoint management, or application operations. Keep ZDX as the source of experience telemetry, then route the validated evidence to the team that owns the failing segment.

Identifying Process-Level Issues

Process-level issues are a frequent trap because they can make a healthy application look slow and a healthy network look guilty. A browser, collaboration tool, endpoint agent, updater, or background service can consume CPU, memory, disk I/O, or network resources in a way that degrades user experience before traffic ever leaves the device. ZDX helps administrators determine whether the endpoint is the source of the user’s pain or merely standing near the crime scene.

Process Inventory provides an organization-level view of processes that might be affecting user devices. It tracks process behavior against a configured CPU usage threshold, updates

calculations in one-minute rolling intervals, and displays top processes periodically. This is useful when the question is not “what happened to this one user?” but “is the same process causing a pattern across many devices?”

The Process Inventory table can show process names, impacted devices, CPU incidents that exceeded the configured threshold, and average memory usage. The process details view can add device-level information such as CPU incidents, average memory, disk reads, disk writes, network upload, operating system, and device vendor. This lets an administrator distinguish a single-device issue from a version, hardware, OS, or software distribution pattern.

Deep Tracing gives a more immediate view for a specific troubleshooting session. In a Deep Tracing result, Top Processes can show which processes consumed the most CPU, memory, disk, or network resources during the captured window. This is particularly valuable when a user says the issue is happening “right now.” Process Inventory provides fleet context; Deep Tracing provides session context.

The process evidence must be correlated with the symptom. A process spike is meaningful when it aligns with the poor ZDX Score, increased Page Fetch Time, degraded device health, user device event, or reported issue window. If the spike occurs outside the affected time range, it may be noise. If the process is busy but the user’s experience metrics remain stable, the administrator should keep investigating rather than forcing the endpoint theory.

Software Inventory and Software Patch Inventory add change context. If a process issue begins after a software update, security patch, or new version rollout, software and patch inventory can help identify the affected users, devices, versions, install dates, and version history. This is how an administrator moves from “one laptop is slow” to “a specific software version appears to be causing CPU incidents across a population.”

Device Inventory adds hardware and platform context. If the same process affects only devices with a specific model, OS, CPU class, or memory profile, the root cause may be capacity, compatibility, or lifecycle related rather than application or network behavior. That distinction matters because the remediation might be hardware refresh, software rollback, policy adjustment, or endpoint engineering escalation.

> ⚠️ **Warning**
> *Top process does not mean guilty process*
>
> A top process can be normal under load. Treat it as a root-cause candidate only when it aligns with user impact, device health degradation, and the affected time window.

Validating AI-Generated Insights

Validation is the step that turns an AI-generated recommendation into an administrator-grade conclusion. If ZDX AI suggests DNS latency, validate it with DNS Resolve Time and affected

timing. If it suggests application responsiveness, validate it with Server Response Time, Page Fetch Time, and Availability. If it suggests path latency or packet loss, validate it with Cloud Path graphs, Hop View, Command Line View, and impacted scope. If it suggests endpoint health, validate it with Device Health, Top Processes, Device Events, and inventory evidence.

Analyze Score can identify likely factors behind a poor ZDX Score and provide a confidence level based on similar probe patterns. The administrator should read the factor, explanation, and confidence level, then inspect the supporting telemetry. Confidence is useful, but it is not the same as certainty. “High confidence” still needs a graph that agrees with it.

Comparison is one of the strongest validation techniques. Comparing a degraded point with a last known good score or a prior time point can show what changed in web, device, and Cloud Path metrics. This is often more useful than staring at a single poor-score moment. In scenario terms, comparison helps identify whether the important change was DNS time, server response, path latency, packet loss, Wi-Fi behavior, or endpoint resource pressure.

Candidates should also know when automated score analysis is not the right tool. For call-quality applications, the troubleshooting evidence may depend on meeting metrics, MOS-related indicators, latency, jitter, loss, and participant-specific call context rather than ordinary web score analysis. The correct approach is to use the data type that matches the application experience being investigated.

Network validation requires careful interpretation. Intermediate hops can rate-limit or deprioritize traceroute-style responses, so packet loss shown at an intermediate hop is not automatically proof of user-impacting loss. Loss at the destination, loss that continues across later hops, latency concentrated in a path leg, and matching application performance symptoms are stronger evidence. The destination and downstream behavior carry more weight than one grumpy middle hop.

Application validation should separate availability from responsiveness. Availability problems point toward failed responses, timeouts, or response codes outside the expected success criteria. Server Response Time points toward time to first byte and possible application-side responsiveness. Page Fetch Time reflects the total fetch experience for the top-level document and can be influenced by multiple factors. The administrator should identify which application metric moved and which stayed normal.

Device validation should combine real-time and historical context. Device Health can show CPU, memory, disk, Wi-Fi, battery, and network indicators for the affected period. User Device Events can explain changes such as network shifts or restarts. Device Inventory, Software Inventory, Software Patch Inventory, and Process Inventory can determine whether the same pattern exists across similar devices or versions. A single spike is a clue; a repeated fleet pattern is a case.

Escalation should be based on validated ownership. If evidence points to a user device or process, escalate to endpoint operations with session and inventory proof. If evidence points to

Wi-Fi or Last Mile ISP, provide affected users, locations, signal or path data, and timing. If evidence points to a Service Edge path, ZPA segment, or application response problem, provide the relevant path, probe, and score context. The cleaner the evidence package, the fewer meetings it takes to stop arguing about whose problem it is.

Automation and ticketing workflows should use validated signals, not raw AI suggestions. Alerts, webhooks, and ITSM routing are valuable when the criteria, thresholds, affected population, and notification path are tuned to real operational patterns. If a workflow triggers on unvalidated or overly broad conditions, it can create alert fatigue and misrouted tickets. Good automation should make the right work appear in the right queue with enough evidence to act.

When evidence is weak, the administrator should say so. A defensible conclusion might be "probable local Wi-Fi issue based on signal degradation and user device events, but no Diagnostics session captured the exact failure window." That is better than pretending certainty. The next action is to collect the missing evidence, tune the monitoring scope, or reproduce the issue under Diagnostics.

🎓 Exam Note

Validate, then act

Scenario answers often test whether the administrator should remediate immediately, collect more evidence, compare against a known-good state, or escalate. If the AI-generated insight is not corroborated, validation is usually the right next move.

📝 Quick Review

- Begin troubleshooting by determining scope: user, device, site, geography, ISP, application, or multiple applications.
- Use ZDX Score as a triage signal, not as the final root cause.
- Correlate Web probe metrics, Cloud Path metrics, device health, process evidence, inventory data, and incident scope in the same time window.
- Use Copilot to accelerate investigation, summarize data, and identify trends, but validate its output before acting.
- Use Deep Tracing through Diagnostics when standard dashboards do not provide enough granularity.
- Select the Diagnostics session type based on suspected failure domain: Deep Tracing, Hi-Fi Cloud Path, Bandwidth Test, or Packet Capture Probing.
- Treat top processes as candidates, not conclusions.
- Use Software Inventory, Software Patch Inventory, Device Inventory, and Process Inventory to determine whether endpoint issues are isolated or systemic.
- Be cautious with intermediate-hop packet loss; destination and downstream corroboration matter more.
- Use comparison against a known-good state to identify what changed.
- Route escalation based on validated ownership, not ticket folklore.
- Avoid automation that acts on unvalidated or overly broad AI-generated conclusions.

🏅 Exam Blueprint Alignment

Scenario Review

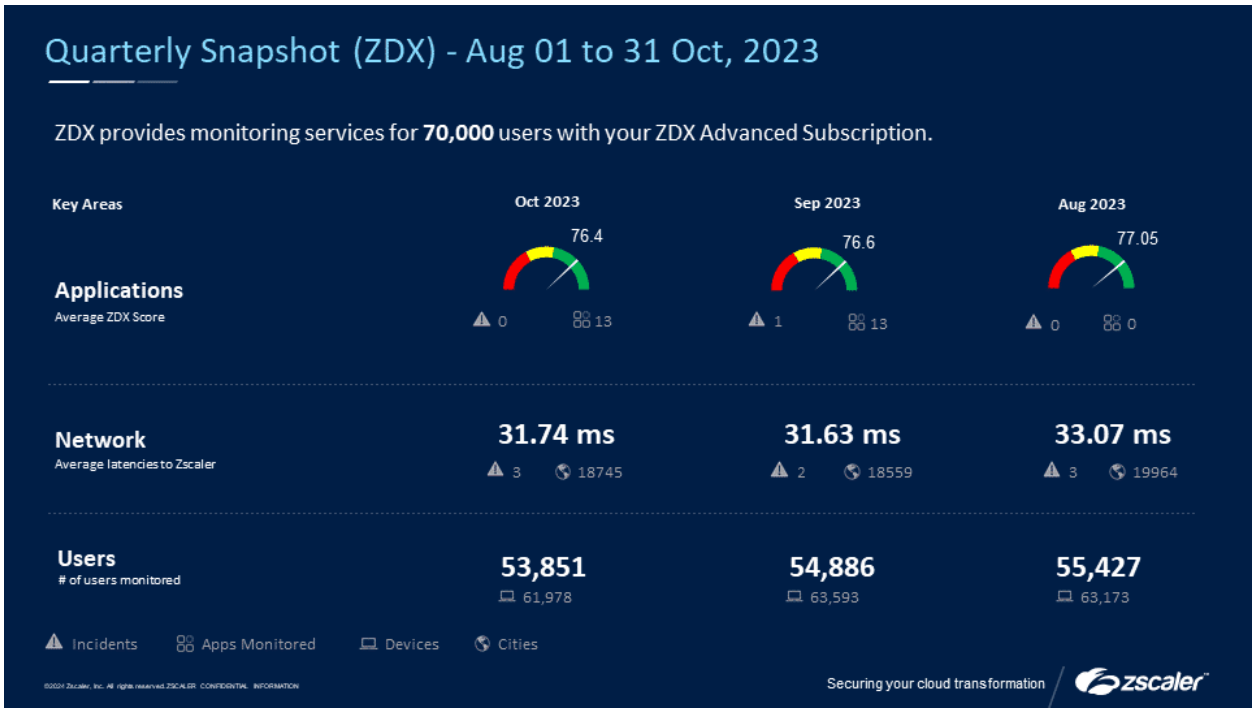
- 3.1 Analyze device, network, and application telemetry to find performance issues across those tiers.
- 3.2 Analyze Diagnostics session output to determine root cause and select an appropriate remediation or escalation path.
- 3.3 Analyze network path information to find bottlenecks through the entire path.
- 3.4 Use Zscaler AI to analyze data for accelerated time to resolution.
- 3.5 Use predefined and customized reports or inventory-style operational views to analyze organization performance patterns.
- 3.6 Share current state or diagnostic evidence when escalation requires a durable artifact.
- 3.7 Leverage ZDX dashboards for organization-wide incidents across categories such as Device, Wi-Fi, Last Mile ISP, ZIA Public Service Edge, ZPA, and Application.
- 1.5 Analyze device health statistics such as CPU, memory, Wi-Fi, disk, and network indicators when validating endpoint-driven degradation.
- 1.6 Use Copilot to compare current performance against historical baselines without manually building custom reports.
- 1.8 Analyze Last Mile or Intermediate ISP performance and identify network anomalies when path evidence indicates network degradation.

Reporting and Data Analysis

Reporting and Data Analysis in this section supports the Troubleshooting and Root Cause Analysis domain by teaching candidates how to interpret structured evidence. The baseline administrator already knows that dashboards and reports exist. The ZDXA candidate must know how to choose between them, how to interpret their time windows and aggregations, and how to avoid drawing conclusions from a view that is too broad for the scenario.

The core reporting workflow is simple but powerful: begin with the symptom, identify the affected object, narrow the affected population, compare against time and peer applications, and then validate with supporting telemetry. That sequence keeps the analysis disciplined. Without it, administrators can easily spend an hour admiring charts while the ticket queue quietly mutates into a management escalation.

Reports are also operational artifacts. A predefined report can support a service review, a Data Explorer view can preserve a recurring troubleshooting lens, a QBR can justify quarterly improvement work, and a snapshot can preserve the current evidence state for escalation. The candidate should think of reporting as both a troubleshooting method and a communication method.



System-Generated

System-Generated reports are standardized analytics views that expose common performance and experience patterns without requiring the administrator to build a custom view first. They are useful when the question maps to a known metric family, such as Cloud Path latency, DNS latency, application performance, Wi-Fi distribution, active users by destination, poor-scoring users, probe assignments, or user location.

Most predefined reports are designed for trend analysis over a rolling reporting window rather than moment-by-moment triage. That distinction matters. The Performance Dashboard is usually the better first stop for an active incident, while System-Generated Reports are better for determining whether a condition has been repeating, expanding, or affecting a specific metric category over time.

Cloud Path End-to-End	Helps analyze average latency across the Cloud Path. In an exam scenario where multiple users report generalized slowness, this report can help determine whether the network path is a meaningful contributor. It becomes more useful when filtered by application, department, location, or geolocation, because broad averages can hide the users who are actually having the bad day.
Last Mile ISP Performance	Useful when the symptom appears concentrated around an access provider or geography. If several users in the same region experience degraded access while the application performs normally elsewhere, this report can support an ISP-focused investigation. It does not eliminate the need to inspect Cloud Path evidence, but it helps administrators decide whether the problem is likely closer to the user's access network than to the application itself.
DNS Performance	Captures DNS latency trends. DNS should be treated as a supporting signal, not a stand-alone root cause label. If Page Fetch Time increases and DNS resolution time also spikes for the affected users or locations, the administrator has a stronger case that name resolution contributed to the experience. If DNS remains stable, the candidate should continue looking at server response, network path, availability, and device data.
Application Performance	Shows the daily distribution of ZDX Scores for the selected application, which helps determine how many users fall into score categories over time.
ZDX Score By Application	Captures average ZDX Score per application, which helps compare monitored applications and identify whether degradation is isolated to one service or visible across multiple services.
Wi-Fi Distribution	Helps identify wireless access patterns.
Active Users by Zscaler Destination	Shows user and device distribution by destination.

Top Users with Poor ZDX Score	Identifies the users most affected for a selected application.
Probe Assignments	Captures the number of enabled probes assigned to users based on probe usage. This report is useful for monitoring design hygiene. Too many assigned probes can create unnecessary measurement overhead and make troubleshooting noisier. In a scenario about excessive monitoring scope or uneven probe assignment, this report helps identify whether probe strategy needs to be adjusted.
User Location Report	Provides an overview of user device geolocations and location history. This report is useful when the administrator needs to validate where users are operating from and whether location context affects score interpretation. If a user's location is unknown, unexpected, or changing, the candidate should treat geolocation evidence as part of the troubleshooting story rather than assuming the user is where the ticket says they are.

The common mistake is treating these reports as a checklist instead of a decision tree. Candidates should choose the report that narrows the next decision. If the scenario is about poor experience for a single application in one country, begin with the application and geography. If the scenario is about multiple applications for users on the same provider, look for network or ISP patterns. If the scenario is about a few poor-scoring users, move toward user, device, Wi-Fi, and Cloud Path evidence.

⚠ Warning

Averages can bury the incident

A healthy organization-wide average can coexist with a painful regional issue. When the scenario names a department, country, location, ISP, application, or user group, filter to that population before interpreting the trend.

Data Explorer

Data Explorer is most valuable when the operational question is specific to the organization. The system-generated reports answer common questions; Data Explorer reporting answers questions such as whether a business-critical application is degrading for a particular department, whether a regional service instance behaves differently from another instance, or whether a recurring ticket pattern lines up with a narrow set of metrics.

A good Data Explorer report begins with a hypothesis. The administrator should decide what the report is meant to prove or disprove before selecting metrics. For example, a report/view designed to compare collaboration application performance should not include every available endpoint metric simply because the controls are there. More data is not always more intelligence; sometimes it is just telemetry confetti.

This is especially useful for repeatable operational reviews. A service owner may need the same set of applications, DNS, Page Fetch Time, and location filters each week. A support lead may need a recurring view of users with poor experience for a critical SaaS application. By preserving the right scope and metric selection, the administrator turns repeated troubleshooting work into a reusable evidence pattern.

The configuration approach should keep the report aligned to one service, one user population, or one operational decision. Mixing unrelated applications or services into the same report can distort the conclusion. If regional application instances behave differently, separating them into distinct reporting views often produces cleaner evidence than treating them as one global service and then trying to explain why the average looks strange.

Data Explorer also requires governance. Views used by service desk, network operations, security operations, or application teams should be owned deliberately. If a view is used for production triage, administrators should avoid broad edit access and should document what the view is intended to support. Least privilege applies to reporting just as much as it applies to policy administration.

Before a particular view becomes part of the operational process, validate it against known dashboard behavior. Confirm the selected time range, filters, applications, metrics, and aggregation. If a custom view disagrees with the Performance Dashboard or a system-generated report, the first assumption should be mis-scoping or aggregation mismatch, not that the platform suddenly decided to freestyle.

🎓 Exam Note

Build Data Explorer views around decisions

A Data Explorer view should answer a decision-oriented question: who is affected, what application changed, which metric moved, what path segment degraded, or what owner should act next. If the report does not support a decision, it is probably dashboard wallpaper.

Data Explorer Usage

Data Explorer gives administrators the flexibility to build and organize customized views of applications and metrics. It is the right tool when the scenario requires correlation that predefined reports do not provide. For ZDXA candidates, the important point is not simply that Data Explorer exists; it is knowing when to use it to build a scoped analytical view.

A Data Explorer view includes several important object references: name, data source, applications, probes when applicable, metrics, aggregation, creator, and last updated information. Each of these fields affects interpretation. The data source tells where the telemetry originates, the applications and probes define what is being measured, the metrics define the signal, and the aggregation determines how the signal is summarized.

Data Explorer can use End User Monitoring data or Hosted Monitoring data. End User Monitoring is appropriate when the question is about the user’s experience from managed devices. Hosted Monitoring is appropriate when the question is about application or service performance from Zscaler-hosted locations. Confusing these sources can lead to the wrong operational conclusion, because a hosted vantage point is not the same as a user device sitting on a home Wi-Fi network behind a creative router from 2014.

Aggregation is a design decision. Average is useful for broad service trends. Minimum can expose the worst user or location experience and is often more relevant when investigating degradation. Maximum can expose spikes that are hidden by average values. In scenario-based reasoning, the candidate should choose aggregation based on the question being answered, not based on habit.

Data Explorer views can be reviewed in chart or tabular format. Chart format is useful for trend recognition and visual correlation. Tabular format is useful when the administrator needs to inspect specific rows, apply filters, or export data for another workflow. The exam may not ask for a procedure, but it can ask which format or view supports the operational outcome.

Only the Super Admin or the admin who created a configured Data Explorer view can edit or delete that view. This supports view ownership and prevents one team from accidentally changing the evidence lens used by another team. For administrators designing operational workflows, ownership and permissions are part of the reporting design.

Data Explorer should be kept clean. Use descriptive names, avoid mixing unrelated services, and build views around the smallest useful scope. A view called “Regional SharePoint DNS and PFT Trend” is far more useful than “Monday Slowness,” even if Monday Slowness feels more honest before coffee.

🎓 Exam Note

Aggregation can change the conclusion

The same telemetry can tell different stories under Average, Minimum, and Maximum. Choose the aggregation that matches the operational risk: broad trend, worst experience, or peak condition.

QBR and Trend Analysis

Quarterly Business Review reports are long-range operational reporting artifacts. They are not intended to replace live incident triage. Their value is in showing how the environment behaved over a quarter, what changed month over month, and where the organization should focus improvement work.

QBR reports are available across ZDX subscription levels, but generation requires a sufficient active-device population for the required duration. Candidates should recognize this as both a prerequisite and a design consideration. A small deployment or pilot may not yet generate the same QBR evidence as a mature production rollout.

The quarterly snapshot includes Application ZDX Score, Network Average Latency, and Users Monitored. The application score in the QBR context is based on the lowest value each user experienced for that application during the calendar quarter, averaged across the relevant user population. This makes the QBR useful for summarizing experience at scale, but not for replacing point-in-time root cause analysis.

The end-user distribution section helps administrators understand where monitored users are located. This is valuable because workforce distribution changes the meaning of performance data. A service that appears healthy from office-centric assumptions may show very different patterns when the same application is accessed by remote, hybrid, or globally distributed users.

Incident trend sections in the QBR help show whether operational disruption is increasing, decreasing, or changing category. A rising pattern of Wi-Fi incidents suggests a different engineering response than a rising pattern of application incidents or ISP-related incidents. The candidate should use these trends to recommend the next operational focus, not simply report that “incidents happened.”

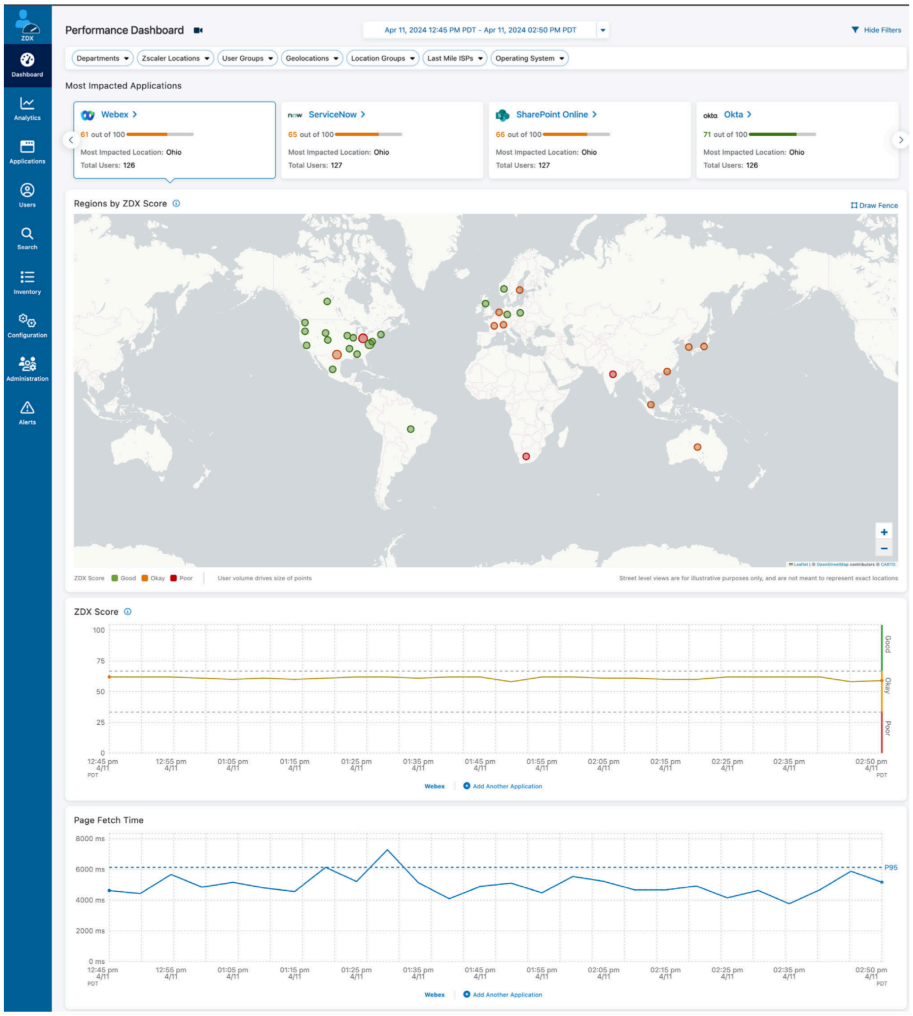
QBR reporting also includes views such as network incidents by network type and ISP, top ISPs by users, application incident overview, application-specific ZDX Scores, average DNS resolution time by location, average latency from source network, alert trends, Wi-Fi analysis, device analysis, and license comparison. These sections provide the evidence needed to justify alert tuning, monitoring expansion, ISP review, endpoint remediation, or platform adoption planning.

QBR reports are downloaded as PowerPoint files and should be used to tell a disciplined service story. The strongest review explains what changed, who was affected, what evidence repeated, what was remediated, and what needs improvement in the next quarter. A weak review says the score was green and moves on. Green is good, but green without context is just a traffic light with a slide template.

Organization-Wide Performance Insights

Organization-wide performance insight is built by correlating dashboards, predefined reports, Data Explorer views, incidents, device evidence, and long-range trend reporting. No single report is authoritative for every troubleshooting question. The administrator's job is to select the smallest set of views that proves scope, likely cause, and next action.

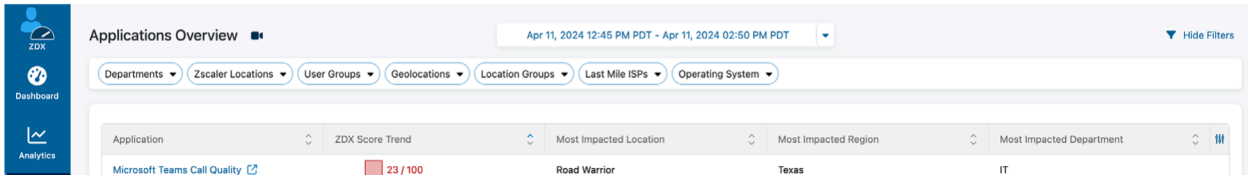
The Performance Dashboard is the usual starting point for organization-wide triage. It exposes impacted applications, regional score mapping, ZDX Score trend, and Page Fetch Time. In a scenario where several users report slowness, this dashboard helps determine whether the issue is broad, regional, location-specific, application-specific, or likely isolated.



The Regions by ZDX Score map is a scope tool. If degradation is concentrated in one geography, the administrator should filter into that region and identify affected users, locations, or departments. If the same application is degraded globally while peer applications remain stable, application-side evidence becomes more important. If several unrelated applications degrade for the same population, network path, DNS, device, Wi-Fi, or Service Edge context should move up the hypothesis list.

Page Fetch Time is one of the most important web application experience indicators because it measures how long it takes the selected application to transfer the fetched page to the user. The P95 line helps expose outlier behavior by showing the percentile boundary for fetch time. A P95 spike with an acceptable average may indicate that a subset of users is experiencing material degradation while the broad population still looks fine.

Applications Overview supports application-centered triage. It helps compare monitored applications, review score trends, inspect impacted departments, regions, and locations, and evaluate probe status. When a scenario names a specific application, this view helps determine whether the problem follows the application everywhere or only appears for a subset of users.



The Incidents Dashboard helps prioritize operational response by grouping detected incidents into categories such as device, Wi-Fi, Last Mile ISP, Internet and SaaS Public Service Edge, Private Applications, or Application. The category should be treated as a strong starting hypothesis. The administrator still needs to validate with impacted users, time range, key metrics, and supporting telemetry before assigning ownership.

Device and Wi-Fi evidence help prevent bad escalations. Device Health can show whether CPU, memory, disk, or network usage is contributing to the experience. Wi-Fi views can expose SSID, access point, signal, or location-specific patterns. If application and Cloud Path evidence look stable but a subset of users shows poor experience, device and Wi-Fi analysis may be the difference between solving the issue and launching the classic blame cannon.

Cloud Path and Network Intelligence help identify last-mile and intermediate network issues. When latency, packet loss, jitter, or hop behavior appears for a population, the administrator should determine whether the issue is near the user, along the ISP path, around a Service Edge segment, or closer to the application. This is the evidence that separates “the app is slow” from “the route to the app is degraded.”

🔄 Cross-policy resources

ZIA, ZPA, Client Connector, and telemetry

- ZDX reporting can intersect with ZIA, ZPA, Client Connector, and platform telemetry. Public applications may traverse Internet and SaaS access paths, private applications may traverse Private Applications paths, and endpoint data may include device, Wi-Fi, tunnel, and process signals. Keep ZDX central, but use adjacent product context when it explains forwarding, identity, path, or endpoint behavior.
- The ZDX API is relevant when reporting data needs to be integrated into external workflows. API access can support reports for applications, users, devices, alerts, inventory, and troubleshooting workflows, depending on the endpoint and permissions. From an exam perspective, API usage should be governed with least privilege, scoped roles, and an understanding of time-window limitations for endpoints that require bounded query ranges.
- Snapshots support evidence sharing during escalation. A snapshot captures the current state of supported UI pages and provides view-only access for review, subject to subscription and role permissions. This is appropriate when another team needs to see the evidence without receiving broad administrative access.
- The final reporting outcome should be a concise operational handoff. State the affected application, affected population, time range, metrics that changed, reports used, and recommended owner. That handoff is the difference between “users say it is slow” and “the affected population is Finance users in one region, Page Fetch Time and DNS latency increased during the same window, peer applications were stable, and the next action belongs with the DNS and application teams.”

🏅 Exam Blueprint Alignment

Scenario Review

- Interpret the Performance Dashboard and ZDX Scores to determine whether the incident scope is single user, site, region, or widespread.
- Use Application view to compare performance across applications and determine whether degradation is application-specific or population-specific.
- Analyze device health statistics when endpoint performance, Wi-Fi, or local connectivity might explain poor experience.
- Analyze Last Mile or intermediate ISP performance and identify network anomalies from path and trend evidence.
- Understand the value of ZDX API access for programmatic reporting, alert review, inventory, and troubleshooting workflows.
- Given predefined and customized reports, analyze organization performance using predefined reports, Data Explorer views, and QBR outputs.
- Share the current state of a UI page through a snapshot when cross-team review or escalation requires a preserved evidence view.
- Leverage ZDX dashboards for organization-wide incidents across categories such as device, Wi-Fi, Last Mile ISP, Service Edge, private application, and application impact.

Snapshots and Shareable Views

A ZDX Snapshot captures the current state of a supported Experience Center UI page and generates a shareable URL for view-only access. The recipient can review a subset of ZDX information without needing to sign in to the Experience Center for that limited view. That makes snapshots useful for service desk collaboration, escalation, post-incident review, and cross-team evidence sharing.

This topic maps directly to the troubleshooting blueprint item that requires candidates to share the current state of a UI page and generate a shareable URL. The exam-relevant skill is not simply knowing that snapshots exist. The candidate must recognize when a snapshot is the right artifact, which page should be captured, what data should be obfuscated, and whether a snapshot is enough or a diagnostics session is still required.

The design intent is shared truth with least-privilege exposure. Snapshot sharing depends on subscription support and the administrator's role permissions. The administrator should treat those permissions as part of the troubleshooting design, not as an afterthought. A user, an application owner, a network engineer, and a service desk analyst may each need different visibility into the same event.

💬 Sidebar
Evidence before escalation

A good snapshot should answer three operational questions: what was in scope, when the issue was observed, and which evidence points to the next owner. If the visible state does not answer those questions, keep investigating before sharing.

Creating Snapshots

Create a snapshot only after the page state has been shaped to match the troubleshooting question. On a user details page, that means the correct user, device, application, time range, and graph context are selected. On an application or incident page, it means filters such as geolocation, location, department, user group, operating system, last mile ISP, or monitoring type have been applied intentionally.

Snapshots can be created from user details, Applications Overview, application details, alert details, diagnostics details, and incident details. Each page serves a different evidence purpose. User details are strongest for single-user or single-device cases. Application details are better for application-centric scope analysis. Alert and incident details help preserve affected populations and triggering conditions. Diagnostics details are better when the investigation has moved into granular session evidence.

For the user details page, snapshot creation is tied to the selected application and selected device. This is a subtle but important exam detail. If a user has multiple devices, the administrator must verify that the correct endpoint is selected before capturing the page. Otherwise, the shared evidence can point the next responder toward the wrong operating system, tunnel type, network adapter, Wi-Fi state, or device performance history.

Use a snapshot name that helps the recipient understand the purpose of the handoff. A name that includes the affected application, region, user case, or incident reference is more useful than a generic label. Keep the name operational, not theatrical. “Finance laptop SharePoint latency handoff” is useful; “mystery slowness again” belongs in the break-room group chat, not in an escalation.

When generating the snapshot, select an appropriate validity period. The available validity window is selected during creation, and it controls how long the shared view remains accessible. Use shorter validity for active troubleshooting and longer validity only when the collaboration or review period requires it. The validity period is an exposure-control decision because the link can continue to provide access until it expires or is deleted.

Obfuscation is the main privacy control during snapshot creation. Depending on the available data types, the administrator can hide sensitive values such as device name, user name, location, IP address, or Wi-Fi name. Choose obfuscation based on the recipient’s need to act. An application owner may need timing and availability evidence, while an endpoint engineer may need device and process context.

After the snapshot is generated, copy the URL and share it only with trusted recipients. The shared view is limited and view-only, but anyone with access to the link can access the captured state. That is powerful for collaboration and risky if handled casually. Treat the link like operational evidence, not like a screenshot in a meme thread.

If automated root cause analysis, score comparison, or diagnostics output is part of the case, run and display that evidence before creating the snapshot. A snapshot taken after the analysis preserves the reasoning context. A snapshot taken before the analysis forces the next responder to infer what the first responder already knew, which is a fine way to make MTTR wear cement shoes.

Before completing the handoff, review the captured state for common mistakes: wrong time range, wrong application, wrong device, stale filters, missing incident context, or unnecessary sensitive data. The recipient will usually trust the page they receive. The administrator’s job is to make that trust deserved.

Snapshots also have a lifecycle. Generated snapshots can be reviewed later, copied again if still active, or deleted when they are no longer needed. Expired snapshots are no longer valid. Deleting a snapshot reduces unnecessary exposure and keeps the collaboration history cleaner for future operational review.

> ⚠ **Warning**
> *Snapshot hygiene*
>
> Do not share a snapshot as a shortcut around analysis. Capture the right page, confirm the selected time window, obfuscate unnecessary identifiers, use the shortest practical validity period, and delete links that are no longer needed.

Sharing UI State

Sharing UI state is more precise than sending a screenshot because it preserves the selected Experience Center view as a limited, view-only artifact. The recipient is not being invited to explore everything in the tenant. They are being shown the evidence state that the administrator intentionally captured.

Choose the page based on the question the next responder must answer. Use an application view when the key question is whether an application is degraded for a population. Use user details when the question is whether one user's issue is tied to device, network, or application evidence. Use alert details when the triggering rule and affected devices matter. Use incident details when the important context is a detected multi-user event. Use diagnostics details when deeper session evidence is already available.

Application-level sharing is useful when the issue needs scope clarification. The Applications Overview and application details views can show the ZDX Score, active users, impacted geolocations, impacted Zscaler locations, impacted departments, and application performance trends. When filtered correctly, that shared state helps another team distinguish between a local complaint and an application pattern that deserves broader attention.

User-level sharing is useful when the investigation depends on the selected user and selected device. The user details view can bring together device details, selected application scores, score-over-time behavior, web metrics, Cloud Path information, device performance, and device events. The administrator should correlate these signals before sharing so the recipient receives a story supported by telemetry, not just a red line on a graph.

Diagnostics detail sharing is appropriate when dashboard evidence is not granular enough. Diagnostics results can include Web Probe metrics, Cloud Path information, device health, top processes, and user device events, depending on how the session was configured. In-progress diagnostics data updates during the session, completed sessions show complete data, and expired sessions do not provide session report data. Capture the diagnostics state when it supports the next action.

Alert and incident sharing helps preserve affected population and triggering context. Alert details can show impacted departments, impacted geolocations, impacted Zscaler locations, expression triggers, impacted devices, and score behavior. Incident details can help

communicate whether the event is associated with categories such as device, Wi-Fi, last mile ISP, ZIA Public Service Edge, ZPA, or application impact.

A shared snapshot has intentionally limited interaction. For example, when viewing a user details snapshot, interactions such as changing the time range, changing filters, selecting another device, selecting another application, or choosing a different root cause analysis data point can be restricted. This limitation keeps the recipient anchored to the evidence state that was shared.

The best handoff combines the snapshot with a concise operational note. The note should state what was observed, what scope was selected, and what action is requested. The snapshot supplies the evidence; the note supplies the reasoning path. One without the other is how cases drift into “who owns this?” territory, and nobody wants that meeting.

🎓 Exam Note

Preserved state versus live investigation

Use a snapshot when the next person needs to see the same state. Use diagnostics when the available evidence is not granular enough. Use live role-based access when the next person must pivot freely across users, devices, applications, filters, or time ranges.

Using Shareable URLs for Collaboration

The shareable URL is the handoff mechanism for the captured state. It can be placed in a ticket, escalation note, collaboration thread, or post-incident record so the recipient can review the same view. Use the URL when the collaboration need is “review this exact evidence,” not “start a new investigation.”

Distribution should follow least privilege. Share the URL only with recipients who have a legitimate role in the case. If a recipient does not need identity-level or network-identifying information, obfuscate it before generating the link. If a recipient needs to act on a specific device or user, leave only the necessary identifying details visible.

In escalation scenarios, a snapshot URL should be paired with a short statement of scope. For example, the note might explain that the selected application is degraded only for one geolocation, that the user’s poor score aligns with a device performance spike, or that diagnostics results show the relevant Cloud Path behavior. This prevents the next responder from treating the snapshot as an unexplained artifact.

Snapshots are useful after automated root cause analysis or score comparison has produced a meaningful result. If the analysis identifies likely contributing factors for a poor score, or if comparison against a last-known-good point highlights key differences, capture that state before escalation. The next responder can then validate or challenge the conclusion from the same evidence.

A shareable URL is not always the best artifact. If the case requires an offline record or formal attachment, use an exported diagnostics report where appropriate. If the next responder must change filters, compare additional users, or investigate outside the captured state, assign the appropriate live role-based access. Use the snapshot when the exact state is the point.

Snapshots are especially useful across time zones and shift changes. The handoff can preserve the state of an incident, alert, diagnostic result, or user investigation at the time the first responder reviewed it. That does not prove causality by itself, but it does preserve the evidence that supported the next step.

If a snapshot was created with the wrong filters, shared too broadly, or generated without needed obfuscation, contain the problem cleanly. Delete the snapshot or let it expire, create a corrected version, and update the handoff note. Do not ask downstream teams to mentally patch a bad evidence object. Replace it.

For the exam, the decision pattern is straightforward. If the scenario asks how to let another party view the current ZDX page state without rebuilding the filters or signing in for that limited view, use a snapshot. If the scenario asks how to discover why the experience is poor, continue with dashboards, root cause analysis, Cloud Path, user details, or diagnostics before sharing.

🔄 Cross-policy resources

Governance and visibility

Snapshot collaboration intersects with role-based administration, diagnostics permissions, alert visibility, privacy controls, and operational governance. The administrator who can view a page is not automatically justified in sharing every visible field. Match the recipient, the page, the validity window, and obfuscation settings to the business need.

📝 Quick Review

- A ZDX Snapshot captures the current state of a supported Experience Center page and generates a view-only shareable URL.
- Use snapshots after narrowing the issue scope, not before the investigation has direction.
- Supported snapshot sources include user details, Applications Overview, application details, alert details, diagnostics details, and incident details.
- On user details pages, verify the selected application and selected device before creating the snapshot.
- Use obfuscation to hide data the recipient does not need, such as user, device, location, IP address, or Wi-Fi details.
- Treat the validity period as a security and governance control.
- Pair every snapshot URL with a short operational explanation of what was observed and what action is requested.
- Use diagnostics or live role-based access when the recipient needs more than the

preserved view.
- Common mistakes to avoid: sharing the wrong time window, capturing the wrong device, omitting obfuscation, sharing with untrusted recipients, and leaving stale snapshot links active.

🏅 Exam Blueprint Alignment

Scenario Review

- Section 3.6: Share current state of UI Page and generate shareable URL. This section maps directly to snapshot creation, supported pages, view-only access, obfuscation, URL sharing, and snapshot lifecycle decisions.
- Section 3.2: Analyze diagnostics sessions output to determine the root cause and implement remediation. This section applies when diagnostics details are captured and shared after granular troubleshooting evidence has been collected.
- Section 3.4: Use Zscaler AI to analyze data for accelerated time to resolution. This section applies when automated root cause analysis or score comparison is run first, then preserved in a snapshot for escalation or collaboration.
- Section 3.7: Leverage ZDX Dashboards for organization-wide incidents across various categories. This section applies when incident or alert details are shared to preserve affected scope, impacted users, triggering conditions, and incident category context.

Organization-Wide Incident Management

Organization-wide incident management begins when an experience issue is no longer just a ticket. The Incidents Dashboard is designed for that transition because it identifies issues that impact multiple users and organizes them into operational categories. Those categories include Device, Wi-Fi, Last Mile ISP, Intermediate ISP, ZIA Public Service Edge, ZPA, and Application.

The exam blueprint expects candidates to recognize organization-wide incident handling as part of troubleshooting, not as a separate monitoring exercise. In a real environment, a ticket may start with “Salesforce is slow,” “Teams is choppy,” or “users in London cannot work.” The ZDXA response should turn that language into a structured question: which users are impacted, where are they, what application or path do they share, and which metrics changed at the same time?

An incident category should be treated as a high-confidence starting hypothesis, not as a completed root cause statement. If ZDX identifies a Wi-Fi incident, the engineer still validates the SSID, access point, connected devices, latency, jitter, and signal evidence. If ZDX identifies an Application incident, the engineer still validates HTTP errors, TTFB-to-PFT relationship, last server leg latency, redirects, and user impact.

The operating model is straightforward: identify scope, validate impact, inspect category-specific metrics, route to the correct owner, communicate clearly, and verify recovery. This prevents scattered escalation and keeps teams from turning a dashboard review into a conference-room spectacle. The data should decide the next action; the loudest ticket should not.

🎓 Exam Note

Incident visibility depends on scope controls

Time range and filters matter. If an expected incident does not appear, adjust the time range, remove overly narrow filters, or apply different filters before concluding that no incident exists. This is especially important when helpdesk reports describe an event that happened earlier than the current dashboard view.

Identifying Widespread Incidents

A widespread incident is identified by correlated impact across multiple users, not by volume of complaints alone. ZDX helps distinguish user noise from operational signal by showing total incidents, impacted users, incidents across key areas, incidents over time, impacted users over time, and incident epicenters. The candidate’s job is to align the reported symptom window with measurable impact in ZDX.

The Performance Dashboard is often the first broad triage view when the symptom is application experience. It shows the most impacted applications, regions by ZDX Score, score trends, and Page Fetch Time trends. When a degradation is visible for one application in one region, the next step is usually to isolate the impacted population rather than jump immediately into endpoint troubleshooting.

The Incidents Dashboard is stronger when the question is, “Is this a bigger event?” It groups incidents by key area and gives a map-based epicenter view. The engineer should use the incident summary to decide whether the impact is concentrated in a category such as Wi-Fi, Last Mile ISP, Intermediate ISP, Service Edge path, private access, device behavior, or application behavior.

Epicenter is an important concept, but it must be interpreted carefully. It represents the center of observed impact for the incident, not always the physical location of the failing component. A Last Mile ISP epicenter can represent users affected through an ISP in a geographic area. An Application incident can span multiple countries because the application is the shared dependency. An Intermediate ISP incident can reflect degradation between providers or within an ISP backbone.

Application incidents need application evidence, not generic slowness language. The useful clues include ZDX Score, HTTP errors, the ratio between Time to First Byte and Page Fetch Time, last server leg latency, and redirect count. If those metrics point to application response behavior across a broad population, the proper next step is application-owner or SaaS-provider investigation, not asking individual users to reboot laptops like it is the universal solvent.

Last Mile ISP incidents and Intermediate ISP incidents require different reasoning. Last Mile ISP issues are closer to the user’s provider relationship and may appear as blackout or brownout conditions. Intermediate ISP issues occur between ISPs or within backbone paths and require path evidence such as internal or peering latency, end-to-end latency, leg latency, packet loss, and DNS resolution time.

ZIA Public Service Edge and ZPA incident categories should be validated with service-path metrics. For ZIA Public Service Edge incidents, look for impacted user score, first-hop and second-hop latency around the Service Edge, ZIA transactions, connectivity errors, Cloud Path probe errors, and Web probe errors. For ZPA incidents, distinguish whether the evidence points to an App Connector path or a ZPA Public Service Edge path.

Device incidents are widespread when many endpoints share the same anomalous behavior. System software hangs and crashes can be analyzed by impacted operating systems, impacted software, CPU usage, memory usage, and event trend. This is a different operational lane than Wi-Fi or ISP troubleshooting, even if the user’s symptom is still “everything feels slow.”

A widespread event should also be evaluated for actual user experience impact. Network Intelligence can detect anomalies, but an anomaly without users experiencing poor ZDX Scores may not require the same urgency as an anomaly that overlaps with a visible poor-score population. The best response weighs both the technical deviation and the user impact.

🎓 Exam Tip
Scope before depth

When a scenario describes multiple users, multiple locations, or recurring complaints over time, start with scope. Use organization and incident views first, then drill into representative users only after you know what population they represent.

Wi-Fi Incident Analysis

Wi-Fi analysis is local-path troubleshooting with wireless context. Because Wi-Fi is often outside central IT control, it can create user experience symptoms that resemble application, ISP, or Service Edge issues. ZDX helps separate those layers by tying user and device Wi-Fi data to access point performance, location, signal, latency, jitter, and score distribution.

The Wi-Fi Dashboard supports practical filters such as SSID, department, Zscaler location, geolocation, operating system, vendor, and access point prefix. The SSID with the highest number of devices is selected by default, which is helpful for quick visibility but can hide a smaller SSID-specific problem. In a scenario, wording such as “only users on one floor,” “only one branch,” “only one SSID,” or “only Windows 11 laptops” should drive filter selection.

Dashboard-level Wi-Fi evidence includes total access points, Wi-Fi performance of access points, total devices connected, ZDX Score distribution, Wi-Fi band distribution, and top geolocations by connected devices. These views help determine whether the problem is broad across a location or isolated to a subset of devices, bands, or access points.

Wi-Fi Performance by Geolocation is useful when the first question is “where is this happening?” Map View helps identify clustered wireless impact across locations, while List View helps compare access points, connected devices, poor-score device counts, and the percentage of devices with poor scores. The engineer should move from map-level visibility to access point-level evidence before routing.

Access point views provide the next layer of precision. Tree View is useful for visualizing access point performance and user volume because tile size reflects user count and color reflects score category. List View is better for operational comparison because it exposes fields such as MAC address, custom name, Wi-Fi performance, user count, device count, average latency, and average jitter.

Access Point Details turn a suspected Wi-Fi issue into routeable evidence. Source-to-gateway latency measures the average time for a packet to travel from the source device to the gateway. Source-to-gateway jitter measures the variation in latency across those packets. The connected-user table can include device, user, BSSID, Wi-Fi adapter, ZDX Score, signal strength, Wi-Fi type, Wi-Fi channel, and operating system.

Privacy and collection settings can create blind spots in Wi-Fi analysis. On certain operating systems, including macOS and updated Windows 11 environments, Client Connector privacy settings may prevent collection of location information for ZDX by default. If Collection Location Info for ZDX is not enabled, SSID and BSSID details may not be captured in the Wi-Fi Dashboard. In an exam scenario, missing Wi-Fi identifiers should trigger validation of collection settings before declaring Wi-Fi healthy.

ZDX can also be configured to identify low-performing Wi-Fi devices by signal strength and retransmission rate instead of ZDX Score. That distinction matters. ZDX Score is experience-oriented, while signal strength and retransmission rate are closer to wireless-layer quality. A strong engineer uses score to understand user impact and wireless metrics to explain the radio condition.

The most common Wi-Fi routing mistake is stopping at “users in this location are slow.” That statement is not enough. The routeable version is “users on this SSID, through this access point or BSSID, show poor scores with elevated source-to-gateway latency, jitter, weak signal, or unfavorable band behavior.” That is the difference between an opinion and a repair queue.

> ⚠️ **Warning**
> *Do not blame Wi-Fi without Wi-Fi evidence*
>
> Geography alone is not proof of a Wi-Fi incident. Confirm SSID, BSSID, access point performance, signal strength, latency, jitter, channel, band, retransmission behavior, or connected-device impact before routing to wireless operations.

Self-Service and Helpdesk Workflows

Self Service is a containment layer for predictable user-side issues. When enabled, it can notify users about device and Wi-Fi problems and provide a short diagnosis with a recommended action. This helps users resolve some issues without opening a ticket and gives the helpdesk additional evidence when a ticket is still created.

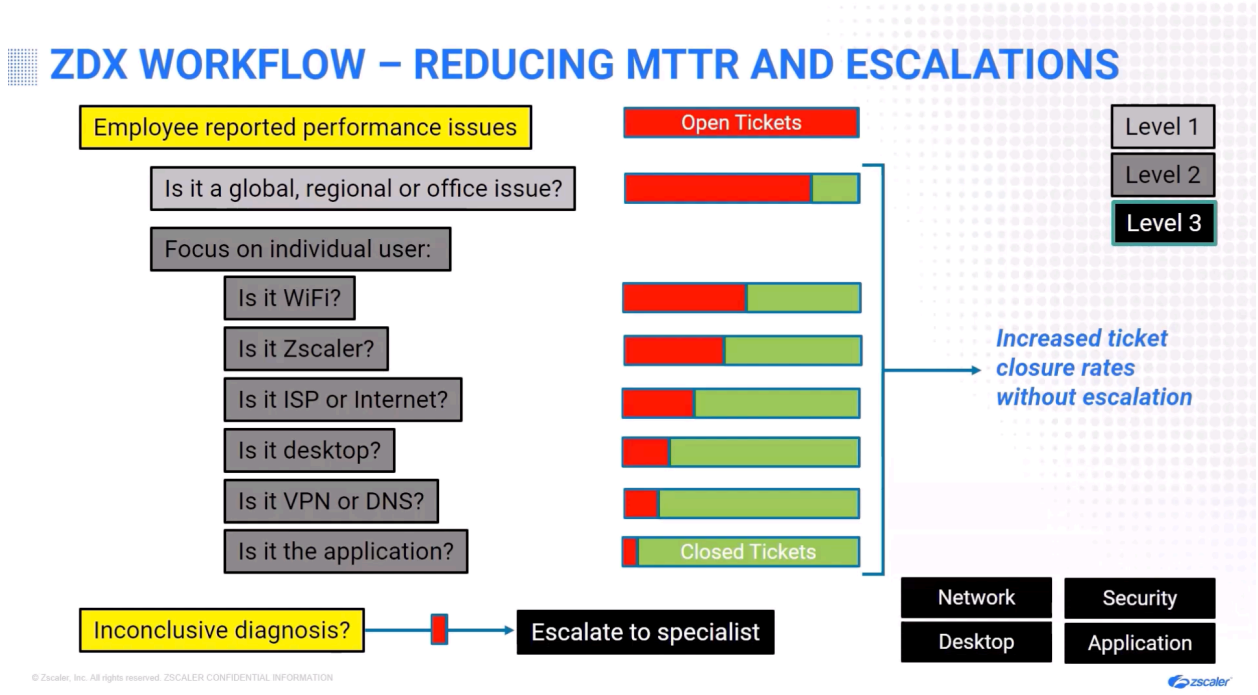
The Self Service Dashboard consolidates notification data so support teams can evaluate both user impact and adoption. Key summary indicators include total notifications sent, notifications by type, total users notified, users who found notifications helpful, active users with Self Service enabled, and users who disabled notifications. These counts help determine whether Self Service is reducing demand or whether the organization needs tuning, education, or a different workflow.

Filtering matters here as much as it does in incident analysis. Self Service data can be narrowed by time range, departments, Zscaler locations, geolocations, and notification types such as Wi-Fi, CPU, or Other. A cluster of CPU notifications in one department after a software rollout tells a different story from scattered Wi-Fi notifications among remote users.

Notifications Sent Over Time is useful for correlating user-facing events with operational timelines. The associated table lists the user, device, notification type, feedback status when available, and timestamp. A support analyst can pivot from a notification to the related User Device Events timeline, where notification icons mark when the user was notified.

High CPU notifications are a common helpdesk containment scenario. A notification can indicate sustained high CPU usage and identify top processes affecting performance. The user may be advised to close contributing applications or use Activity Monitor or Task Manager to end the process. For the helpdesk, the value is not only the recommendation; it is the named process and timestamped evidence.

Wi-Fi Self Service notifications support several different patterns. Degraded Wi-Fi can point to weak signal, high latency to the access point, low bandwidth, or distance from the access point. Suboptimal access point notifications can recommend connecting to a better AP within the SSID network, reviewing mesh behavior, or toggling Wi-Fi so the device reconnects to a stronger access point. Suboptimal Wi-Fi band notifications can recommend using the 5 GHz band when a better band is available.



Helpdesk teams should treat Self Service as assisted remediation, not as ticket deflection at all costs. One notification may help one user. Repeated notifications across a group, location, SSID, or device model should be treated as operational evidence. The workflow is: let the user resolve simple conditions, but escalate patterns that show population-level impact.

Where ITSM integration is deployed, ZDX can also support service desk workflows by bringing alert details and metrics into the incident management process. The operational principle is least privilege: helpdesk roles should have enough access to view relevant dashboards, user details, snapshots, and diagnostics workflows, but not broad configuration permissions unless their role requires it. Clean routing depends on both good data and appropriate access control.

🔄 Cross-policy resources

Client Connector, endpoint operations, and ITSM

Self Service depends on supported Client Connector and ZDX module versions, subscription support, and role permissions. It intersects with endpoint operations when CPU, process, crash, or software evidence points to device remediation, and with ITSM workflows when alerts or incident details are routed into ticketing systems.

Incident Categorization and Routing

Incident categorization is how ZDX turns telemetry into an operational queue. Each category suggests a first responsible domain, but the metrics decide whether that domain should keep ownership. The goal is to route with enough evidence that the receiving team can act immediately instead of rediscovering the same symptom from scratch.

Device incidents should be routed to endpoint operations with operating system, software, CPU, memory, hang, crash, and event-trend evidence. If many impacted devices share the same software or OS version, the likely next step may be software rollback, patch validation, hardware analysis, or endpoint remediation. Do not route these as generic “network slowness” simply because the user noticed the issue while using a cloud application.

Wi-Fi incidents should be routed with SSID, BSSID, access point, geolocation, connected-device impact, latency, jitter, signal strength, Wi-Fi type, channel, and score distribution. If only one AP is degraded, the wireless team needs that AP. If many APs in one site are affected, local wireless infrastructure or RF conditions become more likely. If only one device is affected, endpoint or local support may be the better first path.

Last Mile ISP incidents should include the affected ISP, geolocation, impacted users, applications, and whether the pattern is connectivity-oriented or performance-oriented. Blackout-style evidence supports a connectivity problem. Brownout-style evidence supports degradation such as DNS latency, leg latency, or packet loss. The routing target is usually network operations, with user communication when the affected population is clear.

Intermediate ISP incidents should be routed with affected path information, internal or peering latency, end-to-end latency, leg latency, packet loss, DNS resolution time, and user impact. These incidents can be operationally tricky because the affected segment may be outside both the user’s local network and the customer’s direct provider relationship. The job of the ZDXA engineer is to provide a defensible path narrative, not to claim ownership of the internet’s emotional state.

ZIA Public Service Edge incidents require Zscaler-aware routing. Useful evidence includes impacted user score, first-hop and second-hop latency around the Service Edge, ZIA transactions, ZIA connectivity errors, Cloud Path probe errors, and Web probe errors. These details help distinguish a user-side access issue from a Service Edge path or cloud-related investigation.

ZPA incidents require private access path separation. App Connector incidents should include App Connector context, ZDX Score, first-hop and second-hop latency around ZPA, and App Connector Cloud Path probe errors. ZPA Public Service Edge incidents should include Public Service Edge latency and Web or Cloud Path probe error evidence. This separation matters because App Connector, Service Edge, and application-hosting issues require different owners.

Application incidents should be routed with application-specific evidence. ZDX Score, HTTP errors, TTFB-to-PFT ratio, last server leg latency, and redirect count help determine whether the application is unavailable, responding slowly, redirecting unexpectedly, or creating excessive page fetch time. When the same application is degraded across a wide population, the application owner or SaaS escalation path is often the correct route.

Incident thresholds and dimensions also matter. Some incident types require a minimum number of impacted devices or users before ZDX categorizes the condition as an incident. The dimensions can include country, application, geohash, ISP, ASN, Service Edge, SSID, device population, App Connector, or data center context. The exam-level takeaway is not to memorize every threshold in isolation, but to understand that ZDX incident classification depends on both impact volume and the dimension that defines the affected area.

After routing, the workflow is not complete until recovery is validated. Re-check incident status, impacted users, ZDX Score trend, category-specific metrics, representative user evidence, and Self Service notification behavior. If the incident is no longer widespread but a few users remain affected, update the scope and route the residual cases as individual troubleshooting rather than keeping the broader incident open.

Snapshots and alert actions can support communication and follow-up. Incident details can be captured for sharing or reporting, and alert setup can be used to watch for recurrence. The engineer should use those options to preserve evidence, reduce repeated explanation, and improve handoff quality across teams.

🎓 Exam Tip

Route by the strongest metric

Use the metric that best identifies the owner. SSID, BSSID, signal, latency, and jitter point toward Wi-Fi. HTTP errors, TTFB/PFT, redirects, and last server leg latency point toward application analysis. Software hangs, crashes, CPU, memory, and impacted software point toward endpoint operations. Leg latency, packet loss, and ASN or ISP context point toward network operations. Service Edge probe errors and hop latency around Zscaler services point toward Zscaler-aware escalation.

📝 Quick Review

- Start with scope: single user, device group, SSID, site, region, ISP, Service Edge path, private access path, or application population.
- Use Performance Dashboard evidence for application and regional scope, then use Incidents Dashboard evidence when multiple users or categories are involved.
- Treat incident categories as hypotheses that must be validated by time range, impacted users, epicenter, and key metrics.
- For Wi-Fi incidents, confirm SSID, BSSID, access point, signal strength, source-to-gateway latency, jitter, band, channel, and connected-device impact.
- For Self Service, distinguish individual containment from recurring population evidence.
- For Device incidents, route with impacted OS, impacted software, CPU, memory, system hangs, system crashes, and event trend.
- For Last Mile ISP and Intermediate ISP incidents, correlate impacted users with latency, packet loss, DNS resolution time, affected legs, and path context.
- For ZIA Public Service Edge and ZPA incidents, use Service Edge, App Connector, latency, transaction, connectivity, Web probe, and Cloud Path probe evidence.
- For Application incidents, validate ZDX Score, HTTP errors, TTFB-to-PFT ratio, last server leg latency, and redirect count.
- Avoid common mistakes: drilling into one user before scoping, ignoring time filters, confusing epicenter with physical root cause, blaming Wi-Fi without wireless metrics, or routing incidents without evidence the receiving team can act on.

🏅 Exam Blueprint Alignment

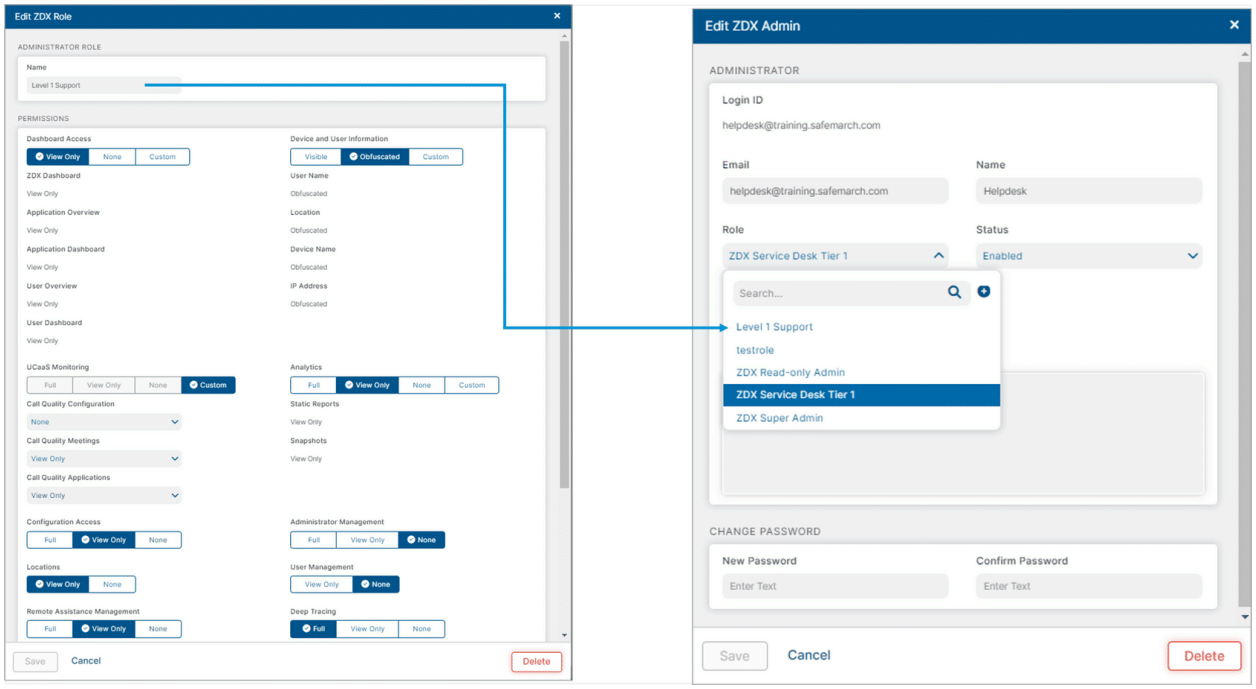
Scenario Review

- Section 1.2: Interpret Performance Dashboard evidence and scores to determine whether impact is single user, site, region, or widespread.
- Section 1.5: Analyze endpoint telemetry, including CPU, memory, Wi-Fi, and device health, to determine whether poor experience is device-driven.
- Section 1.8: Analyze Last Mile ISP and Intermediate ISP performance to identify network anomalies and user impact.
- Section 3.1: Analyze device, network, and application telemetry to locate performance issues across tiers.
- Section 3.3: Analyze network path information to isolate bottlenecks across the full path.
- Section 3.4: Use Zscaler AI and root cause workflows to accelerate analysis and reduce time to resolution.
- Section 3.7: Leverage ZDX dashboards for organization-wide incidents across categories, including Wi-Fi incidents and Self Service workflows.

Administration and Operational Governance (10%)

Role-Based Access Control (RBAC)

Role-Based Access Control is the primary governance mechanism for ZDX administration. It allows organizations to create roles with different permission levels for dashboards, user and device information, UCaaS monitoring, analytics, configuration, diagnostics, administrator management, alerts, webhooks, user management, Client Connector-related access, remote assistance, remediation, self-service, device health, inventory, and other operational areas. Candidates should recognize that RBAC is not a single switch; it is a structured model for controlling both operational capability and sensitive data exposure.



A ZDX administrator assignment is best understood as a combination of role, scope, and data visibility. The role defines the functional permissions. The scope defines the organizational boundary, such as organization, department, application, or location. Visibility controls determine whether user and device attributes are visible or obfuscated. A scenario that asks who should be able to troubleshoot named user tickets, configure probes, manage alerts, or administer other administrators is really asking how these three elements should be combined.

RBAC also supports identity-aligned operations through admin groups when SCIM auto provisioning is enabled. Admin groups can associate a SCIM group with a ZDX role and scope, which allows administrative access to follow the organization’s identity lifecycle. That is especially useful in larger environments where service desk, network operations, security

operations, and platform administration teams are managed through centralized identity processes.

💬 Sidebar

Baseline-to-engineer bridge

At the administrator baseline, candidates learn that roles can be created and assigned. At the ZDXA level, the important skill is explaining why a particular role, scope, and visibility setting is appropriate for a given operational scenario. The exam is more likely to reward containment logic than checkbox bingo.

RBAC Fundamentals

The core RBAC objects in ZDX are administrators, roles, scopes, and, when SCIM is used, admin groups. An administrator is the individual or service identity that receives access. A role defines what that administrator can view, configure, manage, or initiate. A scope limits where those permissions apply. An admin group lets the organization associate SCIM group membership with an administrative role and scope so access can be managed through identity governance.

Role permissions are commonly expressed as Full, View Only, None, or Custom, depending on the feature area. Full allows the administrator to manage or configure the relevant capability. View Only allows observation without modification. None removes access. Custom allows more precise selection inside a permission category, which is useful when a team needs one capability in a feature family but not the entire family.

Dashboard access is usually the safest starting point for operational visibility. A role can allow administrators to view dashboards and overviews without granting configuration authority. This is appropriate for teams that need to understand performance trends, incident scope, user impact, or device health but should not change applications, probes, alert rules, integrations, or administrative settings.

Device and user information permissions deserve special attention because they affect privacy and troubleshooting practicality. User name, location, device name, IP address, and Wi-Fi name can be visible or obfuscated depending on the role design. Obfuscation is valuable when a team only needs aggregate operational insight, but it can limit workflows that depend on user search, device identification, diagnostics, or software inventory exports.

Scope is the second half of the access decision. Organization scope applies across the tenant, while department, application, and location scopes limit the administrator’s operational area. A location-scoped administrator can be designed to operate only within assigned Zscaler locations. A department-scoped administrator can be limited to specific departments. An application-scoped administrator can be aligned to the applications they support.

Scope affects more than what an administrator sees; it also affects what the administrator can configure. For example, scoped administrators can define alert rules, diagnostics sessions, probes, reports, and other operational objects only within the boundaries that their role and scope allow. This is why assigning a broad role with a narrow scope can be safer than assigning a broad role with organization-wide authority.

Some platform settings should remain organization-scoped because they affect the entire tenant. Administrator authentication, integration settings, inventory settings, self-service settings, and other tenant-wide controls should be reserved for platform administrators with a clear governance mandate. If a scenario describes a change that affects all users, all integrations, or tenant-wide collection behavior, narrow departmental access is usually not the right fit.

Admin groups provide a useful way to scale RBAC, but they do not remove the need for role design. SCIM can help populate or maintain administrative group membership, but the group still needs an appropriate role and scope. If the identity group is too broad or loosely governed, SCIM simply automates a bad access model at impressive speed.

Validation is not complete when the administrator can sign in. Validation is complete when the administrator can perform the required tasks, cannot perform unnecessary tasks, sees only the intended data, and is constrained to the intended scope. Practical validation should include testing dashboards, user search, configuration visibility, diagnostics access, alert management, exports, admin management, and scope filters.

> ⚠️ **Warning**
> *Login success is not RBAC success*
>
> A role can authenticate successfully and still be overprivileged, underprivileged, or scoped incorrectly. Always validate the operational outcome, not just the login event.
> Audit logs complete the governance loop. They provide evidence of administrator logins and configuration changes, including useful detail for reviewing what changed before and after a configuration update. In multi-team environments, audit review helps distinguish a legitimate operational change from a mistaken one, and it keeps “someone changed something” from becoming the official incident summary.

Service Desk vs Admin Roles

The Service Desk role exists for fast, controlled triage. Its purpose is to help support teams resolve user-reported issues caused by devices, networks, and applications while reducing escalations, ticket handling time, and user-impact duration. Service desk access should therefore prioritize evidence gathering and user-level investigation, not tenant configuration.

The predefined Service Desk Tier 1 role is intentionally narrow. It includes user name access and user dashboard access, which enables user search and user-focused investigation. That combination makes sense because many support workflows begin with a named user and a complaint such as “SharePoint is slow,” “Teams calls are poor,” or “my applications are laggy.”

A Tier 1 operator should normally be able to identify the user, review the user’s ZDX experience, examine relevant score trends, and decide whether the issue appears isolated or part of a larger incident. The operator should not automatically receive permission to create applications, modify probes, change alert rules, manage webhooks, configure integrations, run broad remediation, or manage other administrators. Triage and tenant control are different jobs.

For Level 2 support, a custom role may be more appropriate than the predefined Tier 1 role. If the support team must run diagnostics sessions, view or export results, inspect additional device evidence, or work within a regional support boundary, the role should grant those capabilities explicitly and pair them with the right scope. The correct design is not “make them Super Admin because they are technical.” That is how tenants develop a trust fall problem.

Data visibility must be aligned with the support workflow. A help desk team that handles named tickets may need visible user names and device names. A regional operations viewer may only need aggregate dashboard access and scoped location visibility. An outsourced team may need obfuscated data unless contract and privacy requirements justify named-user access. The key is to grant the minimum identifiable information needed to resolve the case.

A service desk role should also support escalation quality. The goal is not for Tier 1 to solve every possible root cause. The goal is to collect enough ZDX evidence to route the case correctly: desktop, network, application, security, or platform administration. Good RBAC lets the service desk see the evidence required for that routing decision without giving the team the ability to change the monitoring system itself.

Administrative roles are different because they control platform behavior. An admin role may configure applications, probes, diagnostics, alerts, webhooks, integrations, inventory, self-service, remediation, and other features. Those privileges should be assigned to teams responsible for operating and governing the ZDX tenant, not to every team that consumes ZDX data.

In scenario analysis, look for the verb. If the administrator must “view,” “identify,” “triage,” or “route,” a Service Desk or view-only custom role is often enough. If the administrator must “configure,” “create,” “edit,” “delete,” “manage,” “export diagnostics,” or “administer other admins,” a broader custom administrative role may be required. The verb is the clue hiding in plain sight.

🎓 Exam Tip

Use the narrowest role that completes the workflow

When two role choices both allow the user to complete the task, prefer the role with less write access, less data exposure, and narrower scope. The exam is testing operational judgment, not how quickly you can hand out the crown jewels.

Super Admin Responsibilities

The Super Admin role is the highest-trust administrative role in ZDX. It provides broad read, add, edit, delete, and manage capability across Digital Experience administration. Because of that authority, Super Admin access should be limited to named platform administrators who are responsible for tenant governance, role lifecycle, administrative controls, and high-impact configuration decisions.

During first-time provisioning, the default administrator account is used to establish access and begin creating roles and administrators. That default account is necessary for onboarding, but it should not become the everyday working identity. Named administrator accounts provide better auditability, clearer accountability, and fewer awkward post-change conversations that begin with “who still knows the shared password?”

A Super Admin should own the RBAC lifecycle. That includes creating custom roles, reviewing predefined roles, assigning appropriate scopes, validating administrative access, disabling or deleting obsolete administrators, and ensuring that the role model still matches the organization. Predefined roles, including Super Admin, should be assigned carefully and not treated as ordinary custom roles.

Super Admin responsibilities also include administrator authentication governance. Depending on the tenant design, this can include SAML-based administrator authentication, password expiration for locally authenticated administrators, session timeout configuration, SCIM auto provisioning, and defining what happens when SCIM deletes a linked user account. These settings affect administrative access at the tenant level and should be governed like control-plane configuration.

When SAML is used for administrator authentication, the organization can rely on its external identity provider and associated authentication controls rather than managing a separate local administrator password lifecycle. For local administrator accounts, password requirements and password expiration settings become more important. Candidates should understand the difference because authentication design is part of operational governance, not just login plumbing.

Super Admins should also control administrator management delegation. Giving another administrator Full access to Administrator Management allows that administrator to manage other admins within the limits of role and scope. That may be appropriate for a platform administration team, but it is excessive for most operators who only need ZDX data to resolve incidents.

Audit logs should be reviewed as part of routine governance and after major changes. ZDX audit evidence can show administrator activity, login attempts, and configuration updates, including before-and-after configuration views. This makes it possible to validate expected changes, investigate unexpected behavior, and restore confidence when multiple teams have legitimate access to the tenant.

Remote Assistance is another Super Admin governance area. View-only assistance and full assistance have different risk profiles, and support access should be enabled only for the access level and duration required. A good operational pattern is to grant the minimum support access needed for the case, set an expiration, and verify that assistance is no longer active when the engagement ends.

> ⚠️ **Warning**
> *Super Admin is not a troubleshooting shortcut*
>
> Do not assign Super Admin just because an operator needs to investigate a user issue. Most troubleshooting workflows can be supported with scoped dashboard, user, diagnostics, or inventory permissions. Super Admin should be reserved for administrators who govern the tenant, not everyone brave enough to click “advanced.”

Least Privilege Access Design

Least privilege design starts by defining the work, not by choosing the role name. Identify what the administrator must accomplish: view dashboards, search users, inspect user details, run diagnostics, export results, configure applications, manage probes, create alerts, administer webhooks, manage inventory, configure self-service, or manage other administrators. The role should then grant only those permissions.

Separate read access from action access. Many teams need visibility into ZDX data but do not need to change the tenant. A user experience analyst may need dashboard and analytics visibility. A service desk analyst may need user search and user dashboard access. A network engineer may need visibility into relevant user, network, and Cloud Path evidence. None of those personas automatically require application, probe, integration, or administrator management permissions.

Scope should be as narrow as the operational responsibility allows. Regional teams should be scoped to their locations when the work is location-bound. Departmental support teams should be scoped to their departments when their responsibility is organizationally defined. Application owners should be scoped to their applications when that is the cleanest operational boundary. Organization scope should be reserved for tenant-wide operators and governance functions.

Visibility should be designed persona by persona. If a role only supports aggregate reporting, obfuscate sensitive user and device attributes. If the role supports named-user ticket handling, user and device visibility may be necessary. If the role supports diagnostics, make sure obfuscation choices do not prevent the administrator from accessing the diagnostic workflow or interpreting the result. Least privilege is not the same as making every field invisible; it is granting the minimum useful visibility.

Admin groups should be used where identity lifecycle alignment matters. When SCIM auto provisioning is enabled, an admin group can associate a SCIM group with a role and scope. This is useful when access should follow joiner, mover, and leaver processes. However, SCIM-backed RBAC is only as accurate as the source groups, so identity group hygiene becomes part of ZDX governance.

For API and automation access, use the same least-privilege thinking applied to human administrators. ZDX API usage depends on an API key and an administrator role, so service identities should be dedicated, documented, scoped appropriately, and granted only the permissions required for the integration or automation workflow. Avoid using a broad administrative identity for routine data extraction, reporting, or ticket enrichment.

Alert and webhook administration also requires containment. A team that receives alert notifications may not need permission to manage alert rules or webhook destinations. A team that tunes alert rules may not need administrator management. A team that configures webhook destinations should be trusted to handle integration endpoints and tokens. Split the permissions so the workflow is supported without creating unnecessary tenant control.

Temporary or limited administrative access should be preferred for exceptional tasks. If an administrator needs elevated access for a defined maintenance window or escalation, avoid converting that exception into a permanent role assignment. Where role duration or session timeout controls apply, align them with operational need and review elevated access after the work is complete.

Validation should include negative testing. Confirm that a service desk role cannot modify configuration. Confirm that a location-scoped administrator cannot act outside the assigned locations. Confirm that an obfuscated viewer cannot retrieve sensitive user or device fields. Confirm that a diagnostics-capable role can run the intended sessions but cannot manage unrelated tenant settings. RBAC is only real after the “no” cases are proven.

Access reviews should be scheduled, not improvised. Review Super Admin assignments, Administrator Management permissions, organization-scoped roles, visible user and device data access, diagnostics permissions, webhooks, remediation permissions, and SCIM-backed admin groups. Team changes, reorganization, and project work can all leave behind access that made sense once and now belongs in the museum.

Common least-privilege failures include using Organization scope by default, granting Full access where View Only is enough, giving service desk teams configuration rights, hiding user identity from teams that must search for users, granting diagnostics access without considering privacy controls, and assigning Super Admin because a custom role takes five more minutes to design. Those five minutes are usually cheaper than the incident review.

In exam scenarios, the best answer usually enables the required operational outcome while containing scope, limiting write access, preserving privacy, and maintaining auditability. When a scenario includes multiple administrators or teams, do not think only about who can complete the task. Think about who should complete the task and what access should remain after the task is complete.

🔄 Cross-policy resources

Identity, privacy, and operations

RBAC depends on identity quality, administrative authentication, and privacy requirements. SAML, SCIM, admin groups, obfuscation, audit logs, and administrator lifecycle settings should be treated as one governance system. If those controls are designed independently, the tenant may still function, but the access model will age like milk.

📝 Quick Review

- RBAC combines role permissions, scope, and data visibility.
- Roles define what an administrator can do; scope defines where they can do it.
- View Only is for observation; Full is for management; Custom is for precise delegation; None removes access.
- Obfuscation protects sensitive user and device attributes but can affect user search, exports, and diagnostics workflows.
- Service Desk access should support triage and routing without unnecessary tenant configuration rights.
- Super Admin access should be limited, named, strongly governed, and audited.
- The default administrator account supports onboarding but should not be used for routine daily administration.
- Organization scope is appropriate for tenant-wide governance, not ordinary regional or departmental operations.
- Admin groups can align ZDX administration with SCIM-driven identity lifecycle processes.
- API and automation identities should use dedicated roles and the narrowest permissions required.
- Validate RBAC with both positive and negative tests.
- Common mistakes to avoid: granting Super Admin for convenience, giving Full access to viewers, using Organization scope as the default, over-obfuscating teams that must resolve named tickets, and failing to review access after team changes.

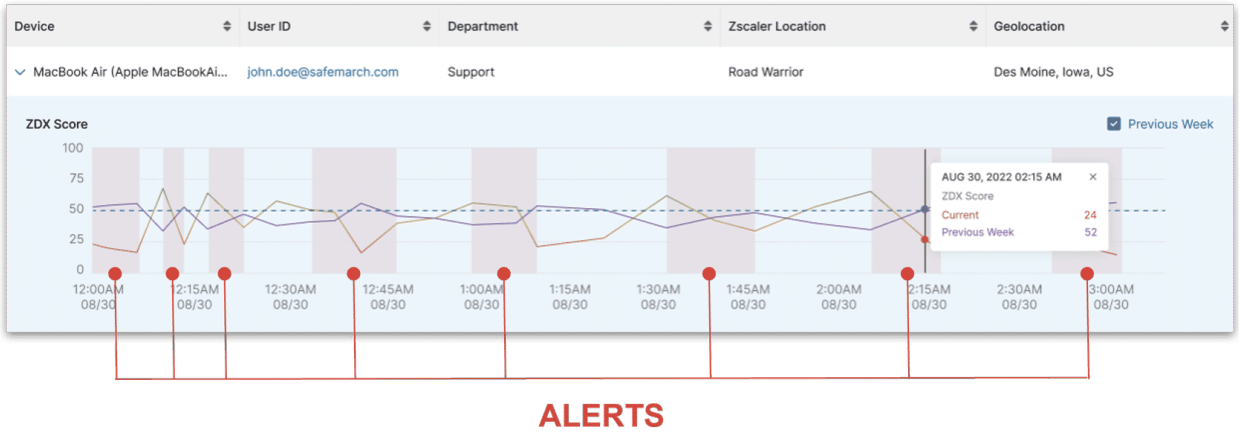
🏅 Exam Blueprint Alignment

Scenario Review

- Section 4.1: Assign role-based access control for a persona, including choosing between Service Desk access, a custom administrative role, and Super Admin.
- Section 4.1: Given a scenario, determine the appropriate role permissions, scope, and user or device visibility settings.
- Section 4.1: Distinguish triage-only access from diagnostics access, configuration access, integration management, and administrator management.
- Section 4.3: Configure administrator governance settings when a scenario requires SAML-based administration, local password governance, session timeout control, SCIM auto provisioning, or administrative lifecycle alignment.
- Section 4.3: Explain why named administrators, appropriate authentication, and audit logs are required for accountable operational governance.

Alerting and Notification Configuration

Alerting and notification configuration maps directly to the Administration and Operational Governance domain because it links ZDX telemetry to operational response. An alert rule defines the conditions that must be met. An alert is the runtime event generated when those conditions are satisfied. Alert details then provide the evidence an administrator or service desk analyst uses to understand impact.



Candidates should think about alerting as a lifecycle. A rule is designed, scoped, given criteria, throttled, routed, reviewed, activated, observed, tuned, and eventually edited or retired. In production, the hard part is rarely creating a rule. The hard part is creating a rule that still makes sense after a pilot becomes a global deployment, after support teams change, and after integrations start creating real tickets.

ZDX alerts can be reviewed in the portal and can also be delivered through email, webhooks, or workflow automation depending on configuration and subscription support. Alert history can be viewed across a selectable time range, and triggered alerts can have a display delay. This matters in operational scenarios because a notification or incident workflow might exist even when an administrator expects immediate visual confirmation in the alert interface.

A mature notification design also includes muted rules. Muting a rule allows administrators to observe alert behavior without sending email or webhook notifications. This is useful during tuning, pilot rollout, and soak periods. It is also a common trap: a muted production rule can make teams believe an important alert path is working when it is only visible inside the portal.

> ⚠️ **Warning**
> *Alerting is not automatically incident management*
>
> An alert proves that configured conditions were met. It does not automatically prove ownership, severity, or root cause. Those outcomes depend on rule design, alert details, notification routing, and how downstream teams consume the evidence.

Alert Rule Design

Start every alert rule with a question. “Is the application slow?” is not specific enough. A stronger design question is, “Do users in this region experience Page Fetch Time or availability degradation for this business-critical application long enough to require service desk action?” That question naturally leads to the correct rule type, metric, scope, threshold, and delivery method.

The rule type narrows the available evidence. Application rules evaluate application-oriented metrics such as DNS Time, Page Fetch Time, Server Processing Time, Web Request Availability, ZDX Score, ZDX Score Drops, and related web performance signals. Network rules evaluate path-oriented metrics such as latency, packet loss, packet count, number of hops, ZDX Score, and ZDX Score Drops. Device rules evaluate endpoint health signals such as CPU, memory, disk, battery, bandwidth, and Wi-Fi signal. Incident rules use incident-specific criteria, and other rule types such as Network Intelligence or Real User Monitoring can apply when those capabilities are in scope.

The best rule type is the one that matches the team expected to respond. Application teams need application evidence. Network teams need path evidence. Endpoint teams need device evidence. Service desk teams need enough context to route or contain the issue without becoming forensic archaeologists in the middle of a busy queue.

Filters define the population under observation. Depending on rule type, filters can include geolocations, Zscaler locations, location groups, departments, user groups, users, and devices. Administrators can use include or exclude logic to narrow the rule, but they should not try to use include and exclude for the same filter dimension. That restriction forces clean scoping and helps prevent contradictory rule definitions.

The scenario-driven skill is deciding what not to monitor. A finance-only application should not generate alerts from users outside Finance. A rule intended for an office outage should not watch the whole organization. A private application accessed through ZPA should be scoped to the users and groups that actually use it, so the rule represents meaningful impact instead of synthetic noise.

Labels and templates are useful governance objects. Labels can be attached to alert rules and then appear in alert details and webhook-delivered alert information, which helps downstream systems group, route, or enrich notifications. Templates can preserve frequently used criteria, especially for hosted monitoring, and help teams avoid handcrafting dozens of inconsistent rule variants.

Current rule design should also treat ZDX Score carefully. If the scenario involves score-based alerting, the cleaner pattern is to configure an Application or Network rule and use ZDX Score or ZDX Score Drops as criteria. That keeps the score tied to the monitored domain instead of floating as an ambiguous tenant-wide symptom.

🎓 Exam Note

Rule type drives the evidence

When a scenario gives you DNS Time, Page Fetch Time, Server Processing Time, Web Request Availability, or Zscaler Time to First Byte, think application. When it gives you latency, packet loss, packet count, or number of hops, think network. When it gives you CPU, memory, disk, battery, bandwidth, or Wi-Fi signal, think device.

Thresholds and Time Windows

Thresholds decide when telemetry becomes an alert. A useful threshold is neither optimistic nor dramatic; it reflects a condition the organization is prepared to investigate. A Page Fetch Time threshold that fires during normal global usage patterns is too low. A threshold that waits until the application is unusable is too high. The administrator's task is to place the threshold where action is justified.

Application thresholds should be tied to user-impacting application behavior. DNS Time points toward name resolution delay. Page Fetch Time points toward the user's perceived page retrieval experience. Server Processing Time and time-to-first-byte style signals point toward server-side or upstream response delay. Web Request Availability points toward reachability or response success conditions.

Network thresholds should be interpreted with path behavior in mind. Latency and packet loss are powerful signals, but packet loss on intermediate hops can be misleading because intermediate routers may rate-limit probe or diagnostic traffic. A network alert is stronger when the path signal correlates with destination behavior, score degradation, or repeated impact across a meaningful population.

Device thresholds should be designed around conditions that can explain user experience degradation. High CPU usage, memory pressure, disk activity, weak Wi-Fi signal, or battery-related conditions can make a healthy application appear slow from the user's perspective. A good device rule helps service desk or endpoint teams identify when the endpoint is the probable failure domain rather than escalating every complaint to the network team.

Time windows and repeated-event settings are what keep alerting civilized. ZDX rules can require a triggering condition to occur multiple times before an alert is sent. This is especially useful for transient spikes that are visible in telemetry but not operationally meaningful. A single bad sample may deserve investigation during a deep trace; it does not always deserve a ticket.

Population thresholds are equally important. Rule actions can use controls such as number of active devices, minimum devices impacted, percentage impacted, and grouping. A scenario where one device in a city violates a threshold should not trigger a city-wide alert if the rule requires a minimum active-device count or a larger affected percentage. This is a feature, not a bug.

Grouping changes the meaning of the same metric. Grouping by city asks whether the condition is significant inside a city. Grouping by organization asks whether it is significant across the tenant. Grouping by department or location can align the alert to the team that owns the affected population. The metric may be identical, but the operational conclusion changes with the group.

Dynamic alerting adds baseline-aware detection through ZDX Score Drops. Instead of relying only on a fixed score threshold, a dynamic alert uses historical baseline behavior and threshold

sensitivity to detect meaningful deviation. This is especially valuable when different regions, applications, or access paths have different normal performance profiles.

> 🎓 **Exam Tip**
> *Thresholds need a population*
>
> When a scenario asks why an alert did not fire, check more than the metric value. Confirm repeated-event count, active-device requirement, minimum impacted devices or users, grouping, filters, rule status, mute status, and whether the alert is still inside any display-delay expectation.

AND/OR Logic in Alerts

Boolean logic determines whether an alert requires all selected criteria to be true or whether any selected criterion is enough. In ZDX rule expressions, All represents AND logic, and Any represents OR logic. This is a small configuration choice with a large operational effect.

AND logic is useful when several signals must line up to prove one condition. For example, a network rule might require packet loss and a poor score condition before notifying the network team. This reduces noise because the alert fires only when multiple pieces of evidence support the same conclusion.

OR logic is useful when each signal is independently actionable. For an application operations team, either Web Request Availability failure or severe Server Processing Time might justify investigation. The rule does not need both conditions to be true because either one can represent a real service issue.

The common mistake is using OR logic as a junk drawer. If unrelated symptoms are thrown into one rule because “we want to know about all of them,” the notification will be difficult to interpret and route. A better design uses separate rules when different symptoms belong to different teams or response paths.

The opposite mistake is overusing AND logic. If a rule requires too many conditions, it might suppress valid incidents because one secondary metric never crosses its threshold. A candidate should recognize scenarios where the rule is too strict and where a simpler expression would produce better operational coverage.

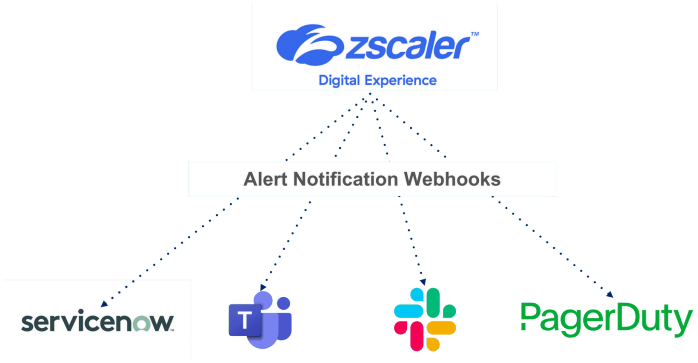
The expression preview is not just a convenience. It is a validation step. Read the expression in plain language before activation: “Alert when all of these conditions are true for this scoped population,” or “Alert when any of these conditions is true for this scoped population.” If that sentence sounds operationally vague, the rule probably is.

> ⚠️ **Warning**
> *Boolean logic can hide the truth*
>
> AND logic can bury real issues when one supporting signal is missing. OR logic can create noise when unrelated symptoms are combined. The correct choice depends on whether the criteria prove the same condition or represent separate actionable conditions.

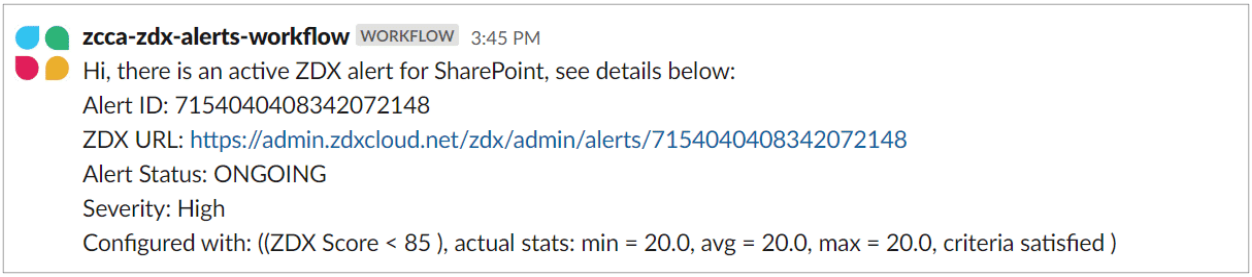
Notification Methods (Email, Webhooks, ITSM)

Notification method should match operational maturity. Email is appropriate for simple awareness, low-volume pilot rules, or administrative visibility. It is easy to configure and easy to understand, but it is not the strongest choice for routing, deduplication, lifecycle tracking, or service-level reporting.

Webhooks are better suited for structured operational workflows. A webhook sends alert data to a provider URL or API endpoint and can be used by multiple alert rules. Webhook configuration includes the destination URL, status, and authentication type, with authentication determined by the provider. The administrator should verify the destination URL carefully because formatting errors can break delivery before the downstream system ever sees the event.



Webhook-delivered alert information can include details such as alert start and end time, impacted devices, severity, rule name, and alert criteria. Labels can add useful grouping context. This is why naming and labeling rules matters: a receiving system can only route intelligently if the payload carries useful context.



ITSM integration is the preferred model when alerts must become owned work. In ServiceNow or a similar system, the goal is not only to receive the alert, but to create or update a record that

can be assigned, categorized, tracked, escalated, resolved, and reported. That turns ZDX alerting into a controlled service management workflow.

Workflow Automation can also participate in alert routing where supported. Administrators should understand delivery-method compatibility. Email can be paired with certain delivery methods, but webhook and Workflow Automation are not selected together as the same delivery path. This prevents overlapping automation paths from competing for the same alert event.

Muted delivery remains an important operational state. A muted rule sends no email or webhook notification, but the alert can still be viewed in the portal. Use this state for validation and tuning. Do not leave important production rules muted unless that is the intended governance decision.

Alert details are the source of truth when triaging. Depending on monitoring type, details can include impacted departments, impacted geolocations, impacted Zscaler locations, expression triggers, average and maximum values, impacted user devices, and status. For hosted monitoring, details can include probe, criteria, start time, end time, and associated metrics such as latency, number of hops, packet loss, or DNS Time.

🔄 Cross-policy resources

Notification routing crosses teams

A ZDX alert may involve ZIA forwarding, ZPA application access, Client Connector telemetry, identity context, endpoint health, ISP path behavior, application performance, or ITSM workflow. Notification design should preserve enough context for the receiving team to act without forcing every escalation to start from scratch.

ServiceNow and Webhook Integration

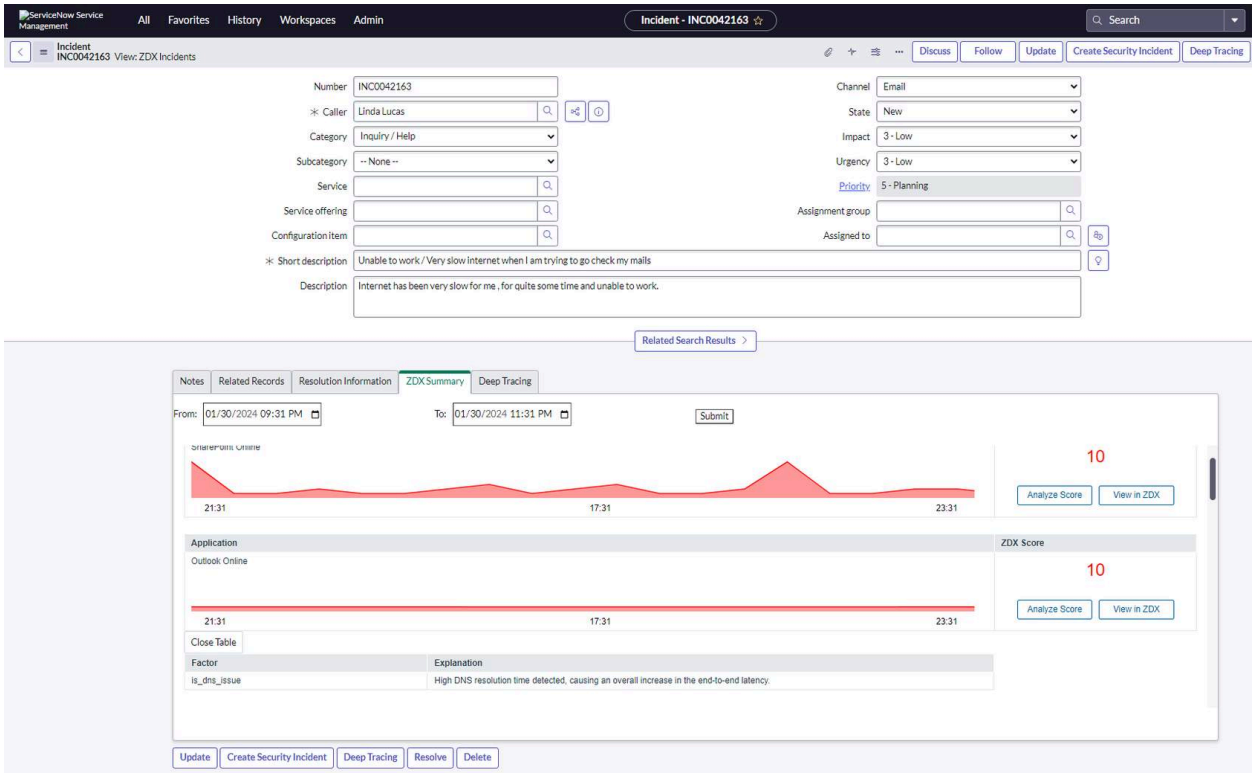
ServiceNow integration should be treated as an operational workflow, not simply a webhook target. ZDX supports integration with ServiceNow for Incident Management and Event Management scenarios, and webhooks can deliver alerts about application, device, or network performance. A successful integration creates records that are understandable, properly categorized, and routed to the team that owns the work.

The ServiceNow integration can also support deeper ZDX operations. With the appropriate Incident Management service user role and ZDX configuration, teams can create Deep Tracing sessions, run Root Cause Analysis for an application score drop, and map categories and subcategories for incoming alerts or created incidents. That reduces tool switching and preserves context for the analyst handling the ticket.

Service user design is a governance decision. ServiceNow should use purpose-built service users rather than personal administrator accounts. Incident Management access, Event Management access, and Deep Tracing access can require different ServiceNow roles. For Incident Management, the ZDX management role supports Root Cause Analysis and alerts, while the Deep Tracing management role supports Deep Tracing operations.

ZDX-side permissions must also match the integration purpose. For alert and webhook administration, the integration administrator needs access to alerts and webhooks, and may need configuration access depending on the workflow. For Deep Tracing and Root Cause Analysis through ServiceNow, the ZDX role must expose the functions and user/device visibility required to locate the impacted user, select the device, and run the diagnostic workflow.

Obfuscation can change what is operationally possible. If user and device information is hidden from a role, some diagnostic workflows may be unavailable or less useful. This does not mean privacy should be weakened by default. It means privacy and troubleshooting requirements must be intentionally balanced when designing service desk and integration roles.



The ServiceNow application settings influence incident creation and lifecycle behavior. Important settings include the service user used for the caller field, logging level, the minimum alert severity that should create incidents, automatic resolution behavior when the source alert ends, the default resolver, ZDX portal and API host values, and API key information where deeper ZDX actions are used. A broken integration scenario may be caused by these settings rather than by the ZDX alert rule itself.

Category and subcategory mapping should be designed with the service desk before production rollout. Device, network, application, and ZDX-related alert types should map to categories that dispatch correctly. If every ZDX alert lands as a generic incident, the integration is technically alive but operationally asleep.

User identity mapping can also matter. If ServiceNow user identities do not match the user identifiers seen from Client Connector, domain settings can be used to append or replace domains for matching purposes. This configuration does not rewrite existing ServiceNow user records; it helps the integration resolve users for basic operations such as fetching ZDX summaries or starting Deep Tracing.

🎓 Exam Note

ServiceNow success is more than a green test

Validate service user roles, ZDX role permissions, authentication, severity filters, category mappings, caller and resolver values, auto-resolution behavior, user identity matching, and whether Deep Tracing or Root Cause Analysis is available to the intended analysts.

OAuth 2.0 Basics for Alerting Integrations

OAuth 2.0 is used when the receiving application requires delegated authorization instead of a static username and password or bearer token. In ZDX webhook configuration, OAuth requires provider-specific application information such as the application selection, client ID, client secret, refresh token expiration, and tenant authentication. The provider determines the authentication model.

At the ZDXA level, candidates should understand the basic roles of the OAuth components. The client ID identifies the application. The client secret proves that the application is authorized. The access token is used to make authenticated calls. The refresh token allows the integration to obtain new access tokens without forcing a human to reauthorize every time the access token expires.

For ServiceNow, creating an OAuth API endpoint is optional depending on authentication requirements, but it is required when the design uses OAuth authentication for the webhook. The ServiceNow endpoint provides the client ID and client secret used by ZDX. Redirect values and tenant-specific host values must align with the ZDX tenant and ServiceNow application configuration.

Token lifetime is an operational tradeoff. A shorter access token lifetime limits the useful life of an exposed token. A longer refresh token lifetime reduces reauthorization overhead. The key design principle is not “make tokens last forever.” It is to set lifetimes that support operations while planning for secure storage, rotation, and revocation.

Testing OAuth webhooks requires a specific expectation. A test operation for OAuth may acquire the OAuth token rather than posting a visible test message in the destination application. If the expected ServiceNow record or collaboration message does not appear immediately, do not assume the webhook failed until you confirm what the OAuth test is designed to do.

OAuth troubleshooting should be methodical. Verify the destination URL, authentication type, application selection, client ID, client secret, refresh token expiration, tenant authorization, provider-side endpoint, and service user roles. Randomly rotating secrets without preserving the original error is the integration equivalent of rebooting the router and calling it root cause.

Secrets must be protected. Client secrets, passwords, bearer tokens, refresh tokens, and API keys should not be placed in screenshots, tickets, shared notes, or study artifacts. For administrator scenarios, the least-privilege answer is usually the one that uses a purpose-built service account, scopes access to the required function, protects credentials, and validates delivery with observable evidence.

⚠️ Warning

OAuth is not a magic webhook checkbox

OAuth adds authorization flow, token management, expiration handling, and provider-side application configuration. If those elements are not maintained, the integration can fail even when the alert rule itself is perfectly valid.

📝 Quick Review

- Design alert rules from operational intent: what condition matters, who is affected, and who owns the response.
- Choose the rule type before choosing the metric. Application, network, device, incident, Network Intelligence, Real User Monitoring, hosted monitoring, and call-quality scenarios expose different evidence.
- Use Application or Network rules with ZDX Score or ZDX Score Drops criteria for score-based alerting.
- Scope alerts carefully with filters such as geolocation, Zscaler location, location group, department, user group, user, or device.
- Do not configure contradictory include and exclude logic for the same filter dimension.
- Use repeated-event settings, active-device requirements, minimum impacted-device thresholds, and grouping to reduce false positives.
- Use AND logic when every criterion must be true to prove one condition.
- Use OR logic when any criterion is independently actionable.
- Muted rules are useful for testing and soak periods, but they do not send email or webhook notifications.
- Email is useful for simple awareness; webhooks and ITSM integrations are better for structured workflows.
- ServiceNow integration should validate roles, authentication, severity filters, category mappings, caller and resolver behavior, auto-resolution, and affected-user context.
- OAuth requires correct provider configuration, client ID, client secret, token handling, tenant authentication, and secure credential management.
- Common mistakes to avoid: overbroad rules, under-scoped filters, noisy OR expressions, overly strict AND expressions, missing throttling, forgotten muted status, using personal admin accounts for integrations, and assuming an OAuth test always posts a visible test message.

Admin Authentication and Security

Admin Authentication and Security connects identity assurance to operational control. ZDX supports administrator access through locally managed administrator credentials and through SAML-based administrator authentication. Where SAML is used, the enterprise identity provider becomes the primary authentication authority for administrator sign-in.

The blueprint emphasis is scenario reasoning. When an administrator cannot complete a task, the correct answer may be an authentication fix, a role permission change, a scope adjustment, a privacy setting change, or recognition that ZIdentity is the source of authority. Treating every access issue as “make the user a Super Admin” is fast, but so is cutting the brakes because the car feels sluggish.

Security also includes the administrative objects themselves. Administrators, roles, scopes, admin groups, password settings, SAML settings, SCIM behavior, session controls, remote assistance, webhooks, and audit logs all contribute to governance. Each one changes either how administrators gain access, what they can do, or how their actions are traced.

A mature ZDX environment uses enterprise identity where appropriate, assigns explicit ZDX roles, constrains those roles with meaningful scope, and obfuscates user or device information when the role does not require it. This keeps operational evidence available without turning sensitive telemetry into casual browsing material.

🎓 Exam Note

Authentication is not authorization

A successful administrator sign-in proves that authentication worked. It does not prove that the administrator can view users, run diagnostics, configure alert rules, manage webhooks, edit roles, or see unobfuscated user and device data. Those outcomes depend on role permissions, scope, subscription support, and privacy settings.

Admin Authentication Settings

Admin Authentication Settings define how administrators enter the ZDX administrative environment and how administrator accounts are governed over time. These settings include password management for locally authenticated administrators, SAML authentication for administrators, SCIM-related lifecycle behavior, session timeout, and administrator account management.

ZDX administrator access can be provided through manually provisioned administrator accounts or through SAML-based authentication with an external identity provider. The local administrator model uses a ZDX administrator login and password. The SAML model relies on the organization’s identity provider so administrator sign-in can align with the organization’s centralized identity controls.

SAML authentication for administrators is more than a convenience feature. It allows ZDX administrative access to inherit enterprise identity practices such as centralized account lifecycle, multifactor authentication, and identity provider policy enforcement. In a design scenario, SAML is usually preferred when the organization wants ZDX access to follow the same identity standards as other privileged administrative systems.

Where local administrator credentials are used, password governance still matters. ZDX administrator password configuration includes complexity requirements when a password is set, and password expiration can be enabled for administrators using the Zscaler-hosted administrator database. This local model should be used deliberately, especially for bootstrap, emergency, or tightly controlled administrative accounts.

Session timeout is another administrative security control. The ZDX portal session timeout determines when an inactive administrator session expires. A shorter timeout reduces exposure from unattended sessions, while a longer timeout may reduce disruption during active incident response. The right setting depends on operational tempo and risk tolerance; the lazy setting is whatever nobody remembers configuring.

When a ZDX administrator is added, the administrator object includes fields such as login ID, email, name, role, status, scope, comments, and password-based login settings when applicable. The role determines functional access. The scope determines organizational reach. The status determines whether the account is enabled or disabled.

Login ID deserves attention because it is not just a display value. It is the administrator's login identifier and cannot be casually overwritten after creation. The cleanest design is to use named accounts that map clearly to individual administrators so audit logs identify the real operator behind a change.

Status is useful for containment. Disabling an administrator can immediately prevent access while preserving the account object for review or later reactivation. Deleting an administrator is a lifecycle action and should be handled as part of offboarding, role transition, or cleanup after confirming that operational ownership has been transferred.

ZIdentity changes where administrators are managed. If the organization is subscribed to ZIdentity, ZDX administrator records may appear read-only in the ZDX administrative view, and administrator configuration is handled from the ZIdentity administrative plane. In a scenario, the correct response is to manage the administrator at the source of authority, not to troubleshoot the read-only state as a ZDX portal defect.

SCIM auto provisioning can be used to support admin group creation. When SCIM auto provisioning is enabled, admin groups can be created based on SCIM group membership and associated with ZDX roles and scopes. This allows administrator access to follow identity group membership instead of relying only on manual per-admin assignment.

Advanced SCIM behavior affects account lifecycle. When a linked user account is deleted through SCIM, the configured action determines whether the associated ZDX administrator

account remains or is deleted. This should match the organization's offboarding policy. For privileged access, "do nothing" and "delete account" are very different answers.

First-time provisioning should be treated as a bootstrap workflow, not as the long-term operations model. The default administrator used to onboard ZDX has broad visibility and authority. That account is necessary to establish initial access, but daily administration should move to named administrator accounts with appropriate roles and scopes so audit trails remain meaningful.

There is also a cross-service credential caveat. If the same credentials are used for a ZDX administrator and an Internet & SaaS administrator, ZDX administrator attributes can affect the corresponding administrative profile, excluding the login ID. During migrations or administrator consolidation, candidates should recognize this as a governance concern, not a trivia footnote.

Validation should include both a sign-in test and a task-level authorization test. After changing authentication settings, confirm that the administrator can authenticate through the intended method, then confirm that the administrator can perform only the intended ZDX actions inside the intended scope. A login-only test is half a test, and half a test is how broad access sneaks into production wearing a support-team hoodie.

> ⚠️ **Warning**
> *Default admin drift*
>
> Do not let the initial default administrator become the normal operating identity. It weakens accountability, complicates audit review, and encourages broad permissions to remain in place long after onboarding is complete.

ZCC Profile Configuration

Not every ZIA or Client Connector user must be a ZDX user. An organization might have a larger population using ZIA and a smaller population enabled for ZDX, depending on licensing, rollout phase, pilot design, geography, department, or operational need. Administrators should understand this as an enablement and entitlement decision, not as an administrator authentication setting.

During implementation, validate that ZDX is enabled for the correct users or groups and that the required Client Connector deployment is in place. If expected users are missing from ZDX views, first confirm that they are actually included in the ZDX-enabled population before troubleshooting dashboards, roles, or telemetry.

This topic should not be overcomplicated. Client Connector is how ZDX collects user-side telemetry, but ZDX administrator access is governed elsewhere through administrator authentication, roles, scopes, and privacy settings. For this section, the exam-relevant point is that ZDX visibility depends on enabling the service for the right users.

🎓 Exam Tip
Keep the boundary clean

If the scenario asks why a user population is not appearing in ZDX, check whether ZDX is enabled for those users or groups and whether Client Connector is deployed. If the scenario asks what an administrator can see or change, check ZDX roles, scopes, and privacy settings instead.

Identity Server Settings

Identity Server Settings are about administrator identity for ZDX. Keep this topic focused on how ZDX administrators authenticate, how administrator accounts are provisioned, and where administrator access is managed. Do not tie this section to Client Connector or endpoint profile settings.

ZDX administrator authentication can be handled with locally managed administrator credentials or through SAML authentication with an external identity provider. When SAML is used, the identity provider handles administrator authentication, while ZDX roles and scopes still determine what the administrator can do after signing in.

If the organization uses Identity Server as the source of authority, administrator management may be handled there rather than directly in the ZDX administrator pages. In that case, ZDX may show administrator records as read-only because Identity Server owns the administrative identity configuration.

SCIM auto provisioning can be used to create admin groups based on SCIM group membership. Those admin groups can be associated with ZDX roles and scopes, allowing administrator access to follow identity group membership instead of relying only on manual administrator assignment.

For exam scenarios, keep the distinction simple: Identity Server settings help determine who the administrator is and how the administrator is authenticated or provisioned. ZDX roles and scopes determine what the administrator can access and where that access applies.

🎓 Exam Tip
Blueprint shortcut

If the scenario is about administrator sign-in, SAML, SCIM, Identity Server ownership, or admin group provisioning, think Identity Server Settings. If the scenario is about permissions, dashboard access, diagnostics, webhooks, obfuscation, or organizational reach, think ZDX roles and scopes.

Securing Administrative Access

Securing Administrative Access begins with persona design. A role should represent a job function, not a person's seniority or persistence in asking nicely. Service Desk, network operations, application operations, security operations, platform administration, and executive reporting personas usually require different combinations of visibility, diagnostics, configuration, and management authority.

ZDX roles define access to functional areas such as dashboards, application views, user views, UCaaS monitoring, analytics, configuration, diagnostics, administrator management, alerts, webhooks, user management, Zscaler Client Connector Portal access, remote assistance, remediation, self service, device health, inventory, and related features. Access can also depend on subscription level, so role design should be validated against the tenant's licensed capabilities.

The primary permission states are Full, View Only, Custom, and None, with privacy controls such as Visible and Obfuscated for user and device information. Full grants management capability for the relevant feature. View Only allows visibility without change authority. Custom allows narrower selections inside a feature area. None removes access to the relevant area.

Predefined roles provide useful reference points. ZDX Super Admin has broad read, add, edit, delete, and manage authority. ZDX Read-only Admin has read permissions. ZDX Service Desk Tier 1 focuses on user name access and the user dashboard, enabling user search and basic user triage. Custom roles should be created when the predefined roles do not map cleanly to the operating model.

Device and user information visibility is a separate security decision from dashboard access. Administrators may need to view experience scores without seeing real user names, device names, IP addresses, locations, or Wi-Fi names. Obfuscation helps support privacy-preserving operations, especially for teams that only need aggregated or scoped troubleshooting visibility.

Obfuscation can also affect workflow capability. If user name visibility is obfuscated, user search and certain software inventory export capabilities can be limited. If device name visibility is obfuscated, software inventory export behavior can also be affected. Candidates should recognize these as intentional consequences of privacy control, not random UI misbehavior.

Diagnostics permissions require special care because diagnostics can expose granular user, device, process, web probe, and Cloud Path information. Full diagnostics access allows management, viewing, and export of diagnostics sessions. View-only diagnostics access is narrower. If user and device information is obfuscated, diagnostics access may not be available because the workflow depends on identifiable user and device context.

Administrator Management is one of the highest-risk permission areas. Full access allows administrators to manage other administrators and roles within permitted constraints. This should normally be reserved for platform owners or tightly controlled administration teams. A

troubleshooting user does not need the keys to the identity kingdom just because they can spell “latency.”

Organization scope carries special weight. Certain settings require organization-level authority, including administrator authentication settings, inventory settings, self service settings, and integration settings such as data collection, SaaS app tenant configuration, and webhooks. In a scenario, if the requested task changes global governance or tenant-wide integration behavior, a department- or location-scoped administrator is the wrong fit.

Scope protects operational boundaries. Administrators can view dashboards, applications, users, inventory, reports, and affected devices based on their assigned scope and permissions. They can configure alert rules, diagnostics sessions, and probes only within the limits of their role and assigned scope. This is what turns “can manage” into “can manage only here.”

Roles and scopes also protect continuity. Alert rules and settings are associated with the administrator’s role and scope rather than depending only on a named administrator account. If an administrator leaves and the account is deleted, the organization can reassign the appropriate role and scope to another administrator instead of recreating the entire access model from scratch.

Audit logs are the accountability layer for ZDX administration. They record administrator activity such as successful and unsuccessful login attempts and configuration changes. Audit views can be filtered by time range, action, category, subcategory, and administrator identity. For configuration changes, pre-configuration and post-configuration comparisons help identify exactly what changed.

Audit review is also a troubleshooting method. If a role behaves differently, an alert no longer fires, a probe configuration changes, or a user gains unexpected visibility, check the administrative trail before assuming a product issue. Humans with change access are still the most reliable source of “unexpected behavior,” right after DNS and the phrase “quick change.”

Remote Assistance Management should be controlled as privileged access. Remote assistance can allow support visibility or change capability depending on how it is enabled. Roles that can manage remote assistance should be limited to administrators who understand the operational and security implications, including expiration and access mode.

Webhooks and alert delivery settings are also governance-sensitive. An administrator with webhook management authority can influence how ZDX events are delivered to external incident-management or messaging systems. Because that affects escalation, visibility, ticket creation, and operational response, webhook management should be assigned only to roles responsible for integrations and alert routing.

Temporary or time-bound access should be considered when available. Some role configurations support time duration choices, which can be useful when elevated access is required for a limited operational window. Permanent broad access should not be the default answer to a temporary incident need.

A secure containment workflow is straightforward. Disable or remove departed or compromised administrators according to lifecycle policy, review recent audit activity, confirm SCIM or ZIdentity behavior aligns with the intended source of authority, and reassign the required role and scope to the replacement administrator. Do not blindly copy a previous administrator's broad access just because "that's how it was set up."

When troubleshooting administrative access, isolate the failure layer. Confirm the administrator exists, confirm the login method, confirm status, confirm role permissions, confirm scope, confirm privacy settings, confirm subscription support, and confirm whether ZIdentity or SCIM is controlling the object. This method is slower than guessing for about six seconds and faster than cleaning up a bad privilege grant for six months.

🎓 Exam Tip

Match persona to permission

For Service Desk, think user search, user dashboard, and controlled user detail visibility. For network operations, think dashboard, application, Cloud Path, diagnostics, and scoped analytics access. For application operations, think application-scoped monitoring and diagnostics. For platform owners, think configuration, integrations, administrator management, audit review, and organization scope.

⚠️ Warning

Scope expansion is not troubleshooting

When an administrator cannot see expected data, first determine whether the data is outside the assigned scope, intentionally obfuscated, blocked by role permissions, controlled by ZIdentity, or unavailable because of subscription or feature constraints. Expanding the administrator to Organization scope should be a governed decision, not the universal wrench.

🏅 Exam Blueprint Alignment

Scenario Review

- Section 4.1: Assign role-based access control by matching administrator personas to ZDX roles, permissions, data visibility, and scope.
- Section 4.3: Configure administrator authentication settings for ZDX, including SAML-based administrator authentication, local administrator credential governance, session behavior, SCIM-related lifecycle behavior, admin groups, and ZIdentity source-of-authority considerations.
- Related governance reasoning: Given an access issue, determine whether the correct remediation is an authentication change, role change, scope change, obfuscation change, identity provisioning correction, or audit review.