FRONTIER.

FRONTIER / NEW ANALYSIS



AWS brings Azure telemetry into CloudWatch Omni, turning observability into a choice of multicloud control point

Amazon Web Services can now collect Microsoft Azure platform logs, metrics, network-flow data and selected traces in CloudWatch Omni. The update advances a broader shift toward cross-cloud operations platforms, while leaving telemetry infrastructure, permissions, governance and costs distributed across both providers.

FRONTIER / Frontier Research / November 23, 2026

NEW ANALYSIS

Built to change a decision.

Amazon Web Services can now collect Microsoft Azure platform logs, metrics, network-flow data and selected traces in CloudWatch Omni. The update advances a broader shift toward cross-cloud operations platforms, while leaving telemetry infrastructure, permissions, governance and costs distributed across both providers.

EDITION **New Analysis**

AUTHOR **Frontier Research**

PUBLISHED **November 23, 2026**

This PDF is a publication rendering of the canonical Frontier News article. The web article remains the live source of truth; analysis, figures, and source citations travel with this edition.

What changed

Amazon Web Services added a managed path for bringing Microsoft Azure platform telemetry into Amazon CloudWatch and CloudWatch Omni on November 21, 2026. The update covers Azure resource logs, Azure Monitor platform metrics, virtual network flow logs and traces sourced from Application Insights, allowing operations teams to examine supported Azure and AWS signals through the same AWS service.[1]

This is a material expansion of CloudWatch Omni rather than the launch of the product itself. Administrators connect a Microsoft Entra ID tenant to an AWS account and Region, then select the Azure subscriptions, regions, resource types, tags and telemetry categories to collect. AWS supports configuration through the CloudWatch Omni interface or through the AWS CLI and AWS-provided setup scripts.[1][2] AWS says collection works across Azure public-cloud regions and can deliver telemetry to every AWS Region where CloudWatch Omni is available; that coverage statement remains a vendor claim rather than a finding from independent product testing.[1]

The connection uses Microsoft Entra ID workload identity federation. AWS documentation says the arrangement does not require CloudWatch to store an Azure secret or key, reducing one credential-management burden. It does not, however, remove the need to authorize and operate a collection path inside Azure.[2] Setup creates or configures Azure resources such as Event Hubs, diagnostic settings and policies, and it can require storage accounts for flow logs. Those resources remain part of the operating architecture even when CloudWatch becomes the principal interface.

The word “traces” also needs qualification. Application Insights AppRequests and AppDependencies records can become spans in CloudWatch, but other records, including AppTraces, arrive as log events rather than spans.[2] The connector therefore does not guarantee complete trace equivalence between the two platforms. It also does not collect telemetry from inside an Azure virtual machine or directly from application code. Those workloads still require separate instrumentation or an agent.[2]

AWS presents the capability as a way to reduce separate monitoring tools, custom pipelines and console switching, including when engineers investigate a request that crosses from an Azure-hosted service to an AWS dependency.[1] Independent coverage has confirmed the announced signal categories, but the available reports rely primarily on AWS disclosures rather than broad production testing.[3][4] Enterprises should therefore treat the published coverage as a starting specification and validate behavior against their own resource mix, trace formats and incident workflows.

Why it matters

The immediate CIO choice is not simply whether to turn on another monitoring integration. It is whether CloudWatch should become the primary operational control point for an estate that includes Azure—and, if so, which telemetry deserves to cross the provider boundary.

A centralized view can reduce the time engineers spend moving among cloud-specific consoles during incidents. It can also make cross-cloud service dependencies easier to follow when an application spans AWS and Azure. Those benefits are most relevant when a shared operations or site reliability engineering team already owns services across both providers. In that environment, routing selected Azure signals into CloudWatch may be more useful than maintaining entirely separate incident views or building another custom collection pipeline. AWS explicitly claims the connector can support this consolidated investigation model.[1]

But centralized visibility is not the same as centralized operations. Azure still supplies and transports the source telemetry, while Microsoft Entra ID governs the federated identity relationship. Event Hubs, diagnostic settings, policies and potentially storage accounts remain on the Azure side.[2] CloudWatch becomes a destination and analysis surface, not a replacement for every source-cloud component. CIOs must assign ownership for both halves of the path: who approves Azure permissions, who maintains the connector, who detects collection failures and who removes Azure resources when a connection is retired.

Cost decisions also become more complicated. Moving telemetry to CloudWatch introduces an AWS ingestion and retention decision while preserving Azure-side infrastructure and associated charges. AWS warns that Azure charges can continue after a CloudWatch connection is deleted unless the Azure resources created for collection are also removed.[2] An enterprise can therefore simplify the operator experience while increasing the number of metered components behind it. The relevant comparison is total cost per useful signal, including source collection, transport, storage, retention and duplicated monitoring—not merely the price shown on the destination platform.

Coverage limits should determine which workloads move first. Platform metrics and resource logs may fit the managed path well, but complete application observability still depends on agents or code-level instrumentation. Likewise, only selected Application Insights records become spans.[2] A team that assumes all Azure traces will appear in CloudWatch with equivalent structure could discover gaps during an incident. Before consolidating dashboards or alerting, architects need a signal-level map showing which records arrive, how they are transformed and which Azure-native views remain necessary.

Identity and governance create another tradeoff. Workload identity federation avoids storing a static Azure secret in AWS, according to AWS documentation, but the connector still receives permissions to access selected subscriptions and telemetry.[2] Security teams need to assess the scope of those permissions, the regions receiving data, retention rules and whether logs or traces contain information subject to internal location or access controls. The managed connection removes some custom integration work; it does not remove the need for cross-cloud authorization and data-governance review.

CIOs should consequently resist treating this update as an automatic reason to eliminate Azure Monitor or an existing independent observability platform. The stronger use case is selective consolidation where CloudWatch already anchors operations, supported resource types cover the target services and cross-cloud incident paths justify moving the data. Organizations with Azure-centered operations, extensive unsupported telemetry or strict limits on cross-provider movement may gain less. The decision should follow operating responsibility and incident topology, not the appeal of a single console.

Frontier take

CloudWatch Omni's Azure support makes one point clear: the strategic layer in multicloud observability is becoming the destination that organizes operational work, not necessarily the cloud that generates the telemetry. AWS is competing to make CloudWatch that destination even when part of the application runs on Microsoft infrastructure.

That changes the architecture discussion. Enterprises have traditionally treated each provider's monitoring service as an attachment to that provider's infrastructure. A managed path for Azure-native platform signals into CloudWatch weakens that boundary. A central operations team can select one environment as its main investigation surface while source-cloud services continue producing and transporting the underlying data. The result is a federated operating layer rather than a genuinely unified telemetry stack.

The distinction matters because consolidation at the interface can conceal distribution underneath. CloudWatch may present Azure and AWS signals together, but identity, diagnostic configuration, Event Hubs, flow-log storage, ingestion and retention remain split across administrative and billing domains.[2] A console can become simpler while the control plane behind it becomes more interdependent. That is an analytical inference from the documented architecture, not a vendor claim.

For most enterprises, the winning pattern will be selective centralization rather than indiscriminate ingestion. Signals that support shared incident response, service-level objectives or cross-cloud dependency analysis have a defensible reason to enter the common operating layer. High-volume telemetry with limited investigative value may not. Nor should a centralized destination receive data that cannot satisfy the organization's location, access or retention policies.

The same discipline applies to tool retirement. CloudWatch Omni can reduce console switching, but the documented limitations prevent CIOs from assuming feature parity with Azure Monitor. Telemetry from inside virtual machines and directly instrumented applications still needs other collection methods, while only specific Application Insights records become spans.[2] Azure-native capabilities may therefore remain necessary for collection, diagnosis or administration even if CloudWatch becomes the first screen used during an incident.

The strongest conclusion is that cross-cloud observability is now an architectural placement decision. CIOs must choose where operators investigate failures, then design the telemetry routes, permissions and cost controls needed to support that choice. CloudWatch Omni expands the credible options for AWS-centered organizations, but it does not erase provider boundaries. It moves those boundaries behind the operational interface, where they require more deliberate engineering and financial governance—not less.

Three moves for CIOs

1. — Choose the control point from real incident paths Map the services, dependencies and operator handoffs involved in recent AWS-to-Azure incidents. Designate CloudWatch Omni as the primary investigation surface only for service groups where the same team owns response across both clouds, then define which Azure signals are required to reconstruct those paths.
   - Decision trigger: Proceed when at least one production service has a recurring cross-cloud dependency and responders currently switch consoles or maintain custom telemetry pipelines to investigate it. Defer broad adoption when Azure operations remain organizationally separate or the required resources are outside documented connector coverage.
   - Why now: The new connector creates a managed alternative to separate views, but value depends on the operating model rather than the number of connected subscriptions. Starting with incident paths prevents a technically successful integration from becoming an expensive, low-use telemetry mirror.[1][2]
2. — Treat the connector as a cross-cloud production system Assign named owners for the Entra federation relationship, Azure diagnostic settings, Event Hubs, policies, flow-log storage and CloudWatch ingestion. Add controls for permission review, collection failure, regional routing, retention and complete decommissioning on both clouds.
   - Decision trigger: Require this operating plan before connecting a production Azure subscription or authorizing telemetry that is regulated, sensitive or expensive to transport and retain. Reassess immediately when permissions, destination Regions or retention periods change.
   - Why now: Managed setup reduces custom pipeline work but still creates infrastructure and billing dependencies in Azure. AWS documents that Azure charges can continue after the CloudWatch connection is deleted unless the associated Azure resources are removed.[2]
3. — Prove signal fidelity before retiring duplicate monitoring Run a bounded production pilot that compares source and destination records for platform metrics, resource logs, VNet flow logs, AppRequests and AppDependencies. Test a known cross-cloud request, document transformations and gaps, and calculate the combined Azure and AWS cost per retained signal before removing any dashboard, alert or native capability.
   - Decision trigger: Retire a duplicate view only after the pilot shows that required signals arrive within the organization’s operational tolerance, preserve enough context for diagnosis and meet access, location and retention policies. Keep Azure-native coverage where VM, application-code or unsupported trace data remains necessary.
   - Why now: CloudWatch converts only selected Application Insights records into spans and does not collect telemetry from inside Azure VMs or directly from application code through this path.[2] A signal-level test turns those documented constraints into an evidence-based consolidation boundary.

Sources

- AWS What's New: Amazon CloudWatch Omni now supports Microsoft Azure platform logs, metrics and traces [1]
- Amazon CloudWatch documentation: Send platform telemetry from Microsoft Azure [2]
- Paras Madan: CloudWatch Omni for Azure — Verify One Cross-Cloud AI Request Before Expanding Ingestion [3]
- Platform Changelog report of the CloudWatch Omni Azure telemetry update [4]

SOURCES / CITATIONS

Trace the argument.

Every source below is linked. Source numbering follows first appearance in the report.

01 **AWS What's New: Amazon CloudWatch Omni now supports Microsoft Azure platform logs, metrics and traces**
https://aws.amazon.com/about-aws/whats-new/2026/11/amazon-cloudwatch-omni-microsoft-azure/

02 **Amazon CloudWatch documentation: Send platform telemetry from Microsoft Azure**
https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/omni-send-platform-telemetry-from-microsoft-azure.html

03 **Paras Madan: CloudWatch Omni for Azure — Verify One Cross-Cloud AI Request Before Expanding Ingestion**
https://www.parasmadan.in/blog/cloudwatch-omni-azure-inference-trace-pilot

04 **Platform Changelog report of the CloudWatch Omni Azure telemetry update**
https://platformchangelog.dev/e/199485-amazon-cloudwatch-omni-now-supports-microsoft-azure-platform-logs-metr

FRONTIER.

TAKE THE NEXT STEP

Push this further.

Research is useful when it changes a decision. Bring us the question, the constraint, or the argument you want tested.

RESEARCH / BRIEFINGS

inquiries@thisisfrontier.ai

Briefings, advisory work, executive discussions, and research inquiries.

REPRINTS / LICENSING

reprints@thisisfrontier.ai

Reprints, enterprise distribution, excerpts, and licensing.

FRONTIER

www.thisisfrontier.ai

Research, Data Insights, News, and the live Frontier corpus.

All rights reserved. Contact Frontier for redistribution or reprint rights.

SEE FURTHER. MOVE SOONER.