64% of organizations believe they should monitor productivity or experience-disrupting endpoints, even if they lie outside their physical control.

44% organizations say learning from incidents (LFI) has the most room for improvement in overall incident management activities.

24% of organizations have breached a contractual service level agreement in the last 12 months

66% of organizations use between 2-5 monitoring or observability tools.

Catchpoint SRE Report 2024

AWS: Your Ally Against Observability Anti-Patterns

Indika Wimalasuriya

Observability - 2024

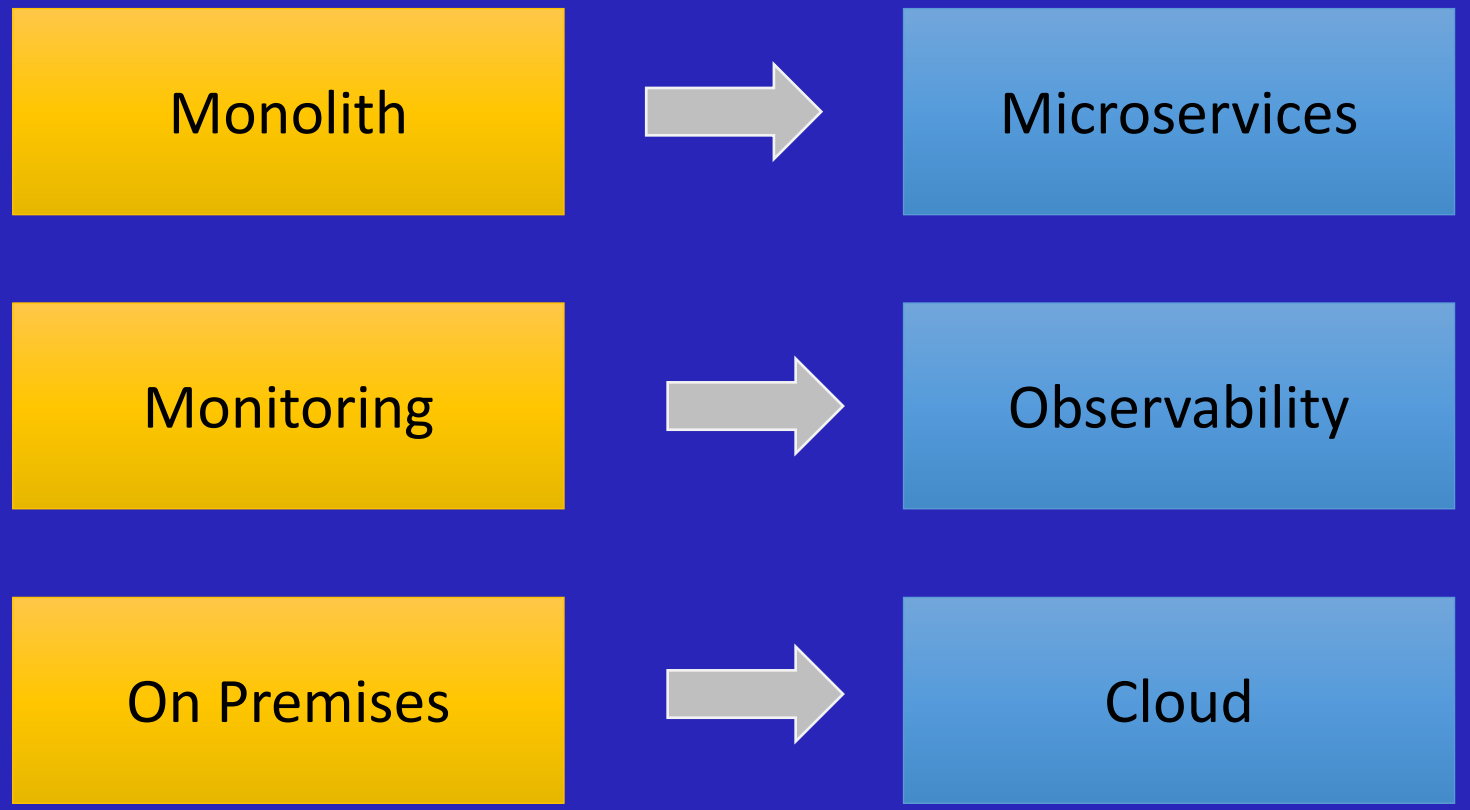
Agenda

- Importance of Observability
- Understanding Observability Anti-Patterns
- Overview of AWS Tools and Services for Observability
- Deep Dive into AWS Observability Tools
- Implementation Guidelines: Eradicating Observability
- Anti-Patterns with AWS
- Best Practices for Observability in AWS

Quick Intro about myself

- Resides in Colombo, Sri Lanka, with my beautiful daughter and wife.
- Reliability Engineering Advocate, Solution Architect (specializing in SRE, Observability, AIOps, & GenAI).
- Employed at Virtusa, overseeing technical delivery and capability development.
- Passionate Technical Trainer.
- Energetic Technical Blogger.
- AWS Community Builder - Cloud Operations.
- Ambassador at DevOps Institute (PeopleCert).

Navigating Digital Transformation: Managing Ever-Growing Complexity



- **Expansion of Data Sources**
- **Surge in Data Volume**
- **Exponential Rise in Failure Scenarios**

Importance of Observability in Distributed Systems

- **Enhanced Performance Monitoring -** Observability provides real-time insights into the performance of various components in a distributed system, helping identify bottlenecks and optimize resource usage.
- **Improved Incident Response -** With observability, teams can quickly detect, diagnose, and resolve issues, minimizing downtime and maintaining system reliability.
- **Increased System Reliability -** By continuously monitoring and analyzing system health, observability helps ensure that distributed systems remain stable and reliable under varying loads and conditions.
- **Better Debugging and Troubleshooting -** Detailed logs, metrics, and traces allow for effective root cause analysis, making it easier to understand complex interactions and dependencies in a distributed environment.
- **Proactive Optimization -** Observability enables proactive identification of potential issues and optimization opportunities, leading to more efficient and cost-effective system operation.

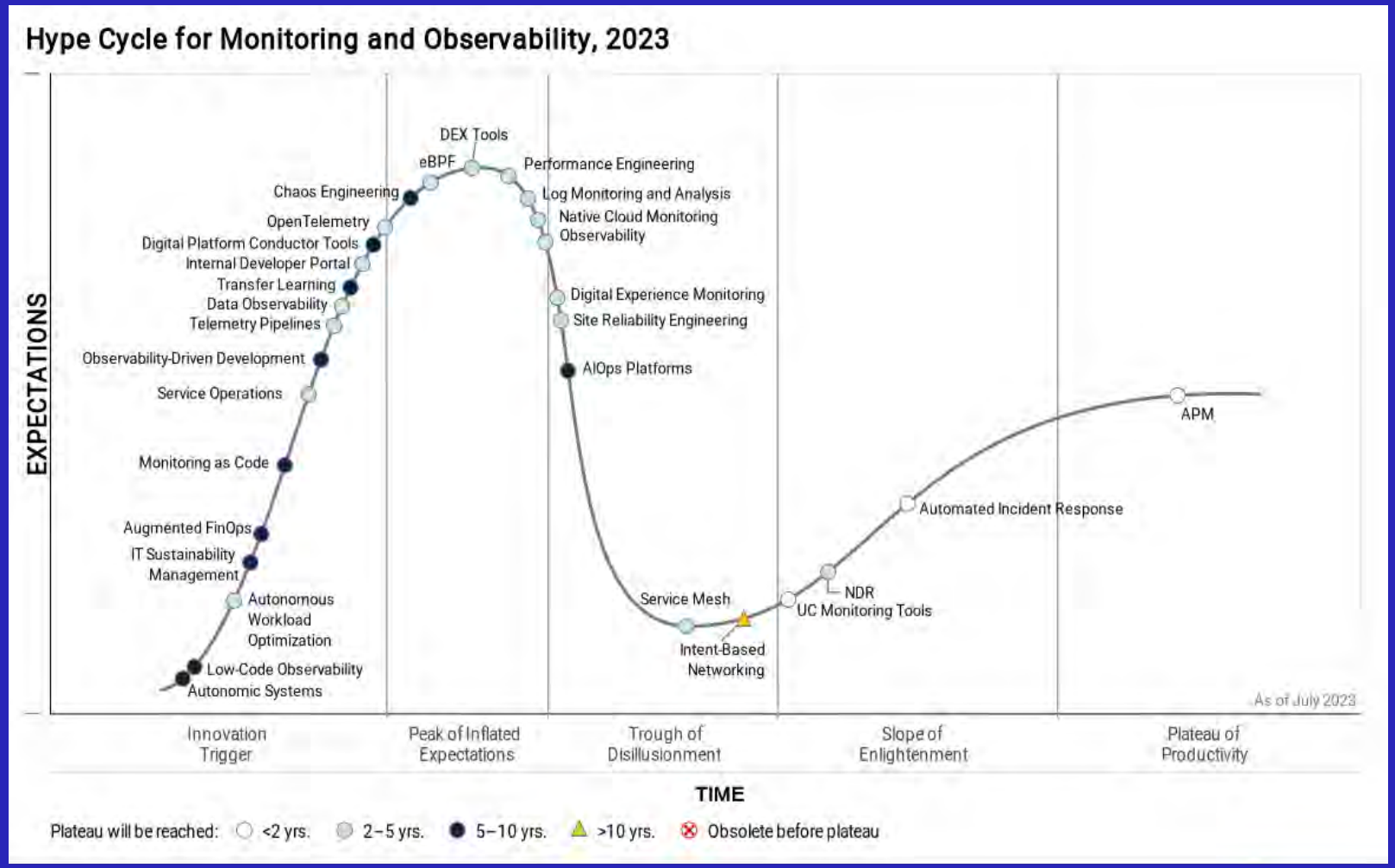
Importance of Observability in Cloud

- **Scalability Management:** Observability helps monitor and manage the dynamic scaling of cloud resources, ensuring optimal performance and cost efficiency.
- **Enhanced Security:** Continuous monitoring detects anomalies and potential security threats, helping to safeguard cloud environments from breaches.
- **Multi-Cloud Visibility:** Provides a unified view across multiple cloud providers, ensuring seamless operation and management of hybrid and multi-cloud deployments.
- **Automated Incident Response:** Enables automation of responses to incidents and performance issues, reducing the time to resolution and minimizing impact.
- **Cost Optimization:** By tracking resource usage and performance, observability helps identify opportunities for cost savings and prevents over-provisioning.

Importance of Observability in world of Microservices

- **Dependency Tracking:** Observability helps monitor interactions and dependencies between microservices, ensuring smooth communication and identifying failure points.
- **Performance Optimization:** Provides insights into the performance of individual microservices, allowing for targeted optimizations and efficient resource allocation.
- **Rapid Issue Detection:** Facilitates quick identification and resolution of issues within specific microservices, minimizing downtime and improving reliability.
- **Scalability Insights:** Offers real-time data on how each microservice scales under load, helping maintain performance during traffic spikes.
- **Enhanced Debugging:** Detailed logs, metrics, and traces make it easier to troubleshoot and debug complex microservice architectures, speeding up development and deployment.

Gartner Monitoring & Observability Hype Cycle



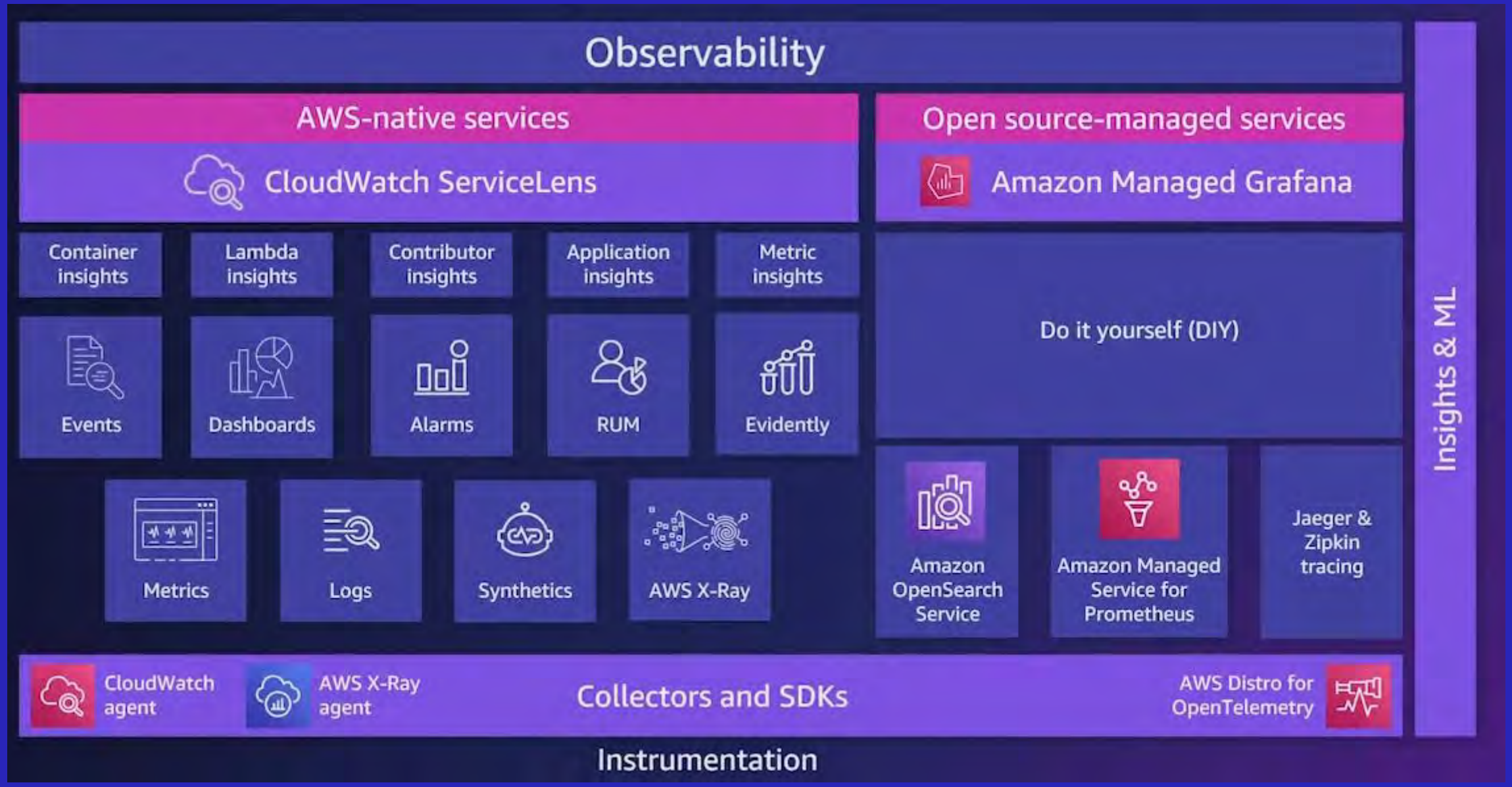
Gartner

Plateau of Productivity: Technologies here have proven their value and are widely adopted. The only example here is:

- **APM (Application Performance Management)**

Observability

- **Logs** - Records of events and activities in your systems and applications. Useful for troubleshooting issues and auditing.
- **Metrics** - Quantitative data about performance and behavior over time. Help track trends and identify anomalies.
- **Tracing** - Follows a request as it flows through distributed systems. Used to analyze bottlenecks and errors.
- **Alarms** - Automated notifications when certain thresholds are breached. Help quickly identify and respond to issues.
- **Dashboards** - Visual representations of metrics, logs and other data. Provide at-a-glance views of system health.
- **Canaries** - Automated tests that run synthetic transactions to monitor availability and performance
- **Real User Monitoring** - Captures performance from an end user perspective. Surfaces issues that affect users.
- **Infrastructure Monitoring** - Monitors the health and utilization of underlying resources like servers, databases, etc.
- **Network Monitoring** - Observes network connectivity and traffic to detect problems and optimize performance.
- **Security Monitoring** - Detection of security threats, anomalies and unauthorized activities.
- **Cost Optimization** - Tracking usage and spending to optimize costs.



Source : Amazon

Logs

Observability Anti-Patterns	Why It's a Problem	AWS Services to be Used	How AWS Can Help
Excessive Logging and Lack of Structured Logging	Generates noise, making it difficult to extract useful insights and identify important events or errors.	Amazon CloudWatch, AWS CloudTrail	Centralizes and manages logs in a structured format, supports various logging frameworks and SDKs for easy integration.

Metrics

Observability Anti-Patterns	Why It's a Problem	AWS Services to be Used	How AWS Can Help
Unclear and Misaligned SLIs and SLOs	Leads to misprioritized efforts and monitoring that doesn't align with business objectives or user expectations.	Amazon CloudWatch Metrics	Defines custom metrics, sets alarms based on SLIs and SLOs, aligns with AWS infrastructure and services, offers SLA commitments.
Bad Sampling Intervals for Metrics	Can result in insufficient data for analysis or an overwhelming volume of traces, impacting performance and observability.	Amazon CloudWatch Metrics	Customizes data resolution and sampling intervals, optimizes resource usage while ensuring adequate data for analysis.
Monitoring Numerous Metrics Unrelated to Customer Experience	Leads to unnecessary complexity and difficulty in prioritizing relevant performance aspects.	AWS Compute Optimizer, Pre-configured AWS monitoring solutions	Focuses on essential metrics directly impacting customer experience, provides optimization recommendations based on resource utilization.

Tracers

Observability Anti-Patterns	Why It's a Problem	AWS Services to be Used	How AWS Can Help
Tracers Are Not Given the Priority They Deserve	Hinders understanding of request flows, making it challenging to troubleshoot performance issues in distributed systems.	AWS X-Ray	Provides distributed tracing capabilities, integrates tracing into applications using X-Ray SDKs.
Lack of Consistent Trace IDs for Distributed Tracing	Disrupts the continuity of distributed traces, making it difficult to follow request flows during troubleshooting.	AWS X-Ray	Ensures consistent trace IDs across AWS services, offers X-Ray Analytics for deep trace data insights.
Not Instrumenting the Code Correctly	Leads to fragmented and decoupled traces, making it difficult to gain a holistic view of system behavior.	AWS X-Ray SDKs	Enables seamless instrumentation of AWS-based applications, produces detailed traces across distributed services.

Tracers (Cont.)

Observability Anti-Patterns	Why It's a Problem	AWS Services to be Used	How AWS Can Help
Over-Instrumentation and Inconsistent Trace Context	Creates unnecessary overhead and disrupts trace continuity, affecting performance and observability.	AWS X-Ray	Configures sampling rates, maintains consistent trace contexts, supports group and aggregate operations for traces.
Long Trace Spans	Makes it harder to pinpoint specific issues and bottlenecks within the system by covering the entire request lifecycle in a single trace.	AWS X-Ray, AWS Step Functions	Breaks down long trace spans into smaller segments, enhances trace granularity and troubleshooting.
Not Unifying Real User Monitoring (RUM) and Application Performance (APM) Data	Results in a fragmented view of user experience and system performance, hindering comprehensive analysis.	AWS X-Ray, APM tools integration	Integrates RUM and APM data for a holistic view, enriches overall observability of applications.

End to End

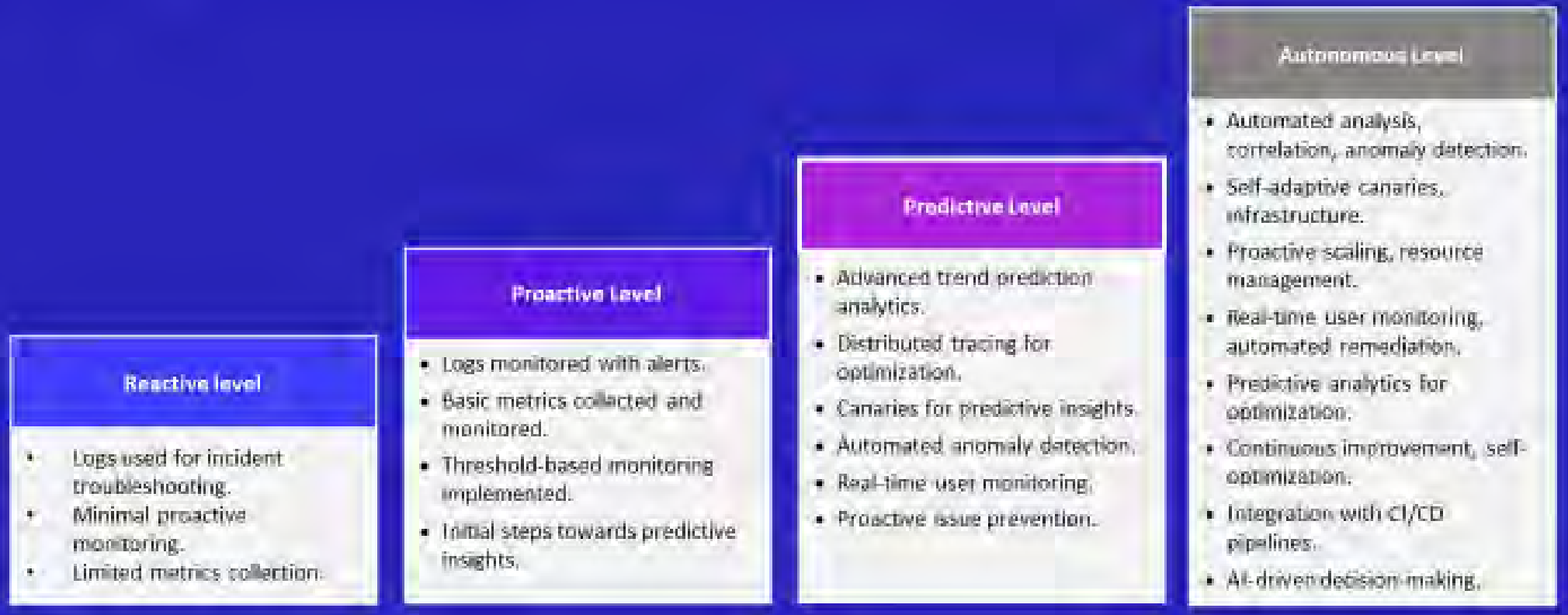
Observability Anti-Patterns	Why It's a Problem	AWS Services to be Used	How AWS Can Help
Alert Overload from Unnecessary Alerts	Causes alert fatigue, making it difficult to distinguish critical incidents from non-essential notifications.	Amazon CloudWatch Alarms, Amazon SNS	Sets up intelligent alerts with custom thresholds and anomaly detection, reduces unnecessary alerts, follows best practices.
Disjointed Observability Tools and Numerous Dashboards	Creates confusion and hinders a unified view of the system's performance, complicating incident response.	AWS X-Ray, Amazon CloudWatch, AWS Personal Health Dashboard	Provides a unified observability platform, offers centralized multi-account management with AWS Control Tower.
Ignoring Non-Functional Requirements	Results in an unstable and unreliable system, affecting scalability, reliability, and maintainability.	AWS Lambda, Amazon RDS	Ensures scalability and reliability with managed services, focuses on non-functional requirements throughout the development lifecycle.

End to End

Observability Anti-Patterns	Why It's a Problem	AWS Services to be Used	How AWS Can Help
Not Understanding the Ecosystem - Upstream and Downstream Impact	Results in blind spots and hinders effective resolution of performance issues by neglecting external service interactions.	AWS CloudFormation, AWS Systems Manager	Manages infrastructure and dependencies better, offers a comprehensive understanding of system interactions.
Environment Inconsistency - Prod, Staging, Test, etc.	Causes unexpected discrepancies and failures, making it harder to reproduce issues and perform reliable testing.	AWS Elastic Beanstalk, AWS CodePipeline	Automates environment provisioning and management, ensures consistency across different environments.

Not Having a Plan is the Biggest Observability Anti-Pattern!

The 4 Stages of the AWS Observability Maturity Model



AWS Pillars from Reactive to Autonomous

Pillars of AWS Observability	Reactive	Proactive	Predictive	Autonomous
Logs	• Logs used for troubleshooting after incidents	• Monitoring logs with alerts for abnormal patterns	• Advanced analysis for trend prediction	• Automated analysis, correlation, anomaly detection
Metrics	• Basic collection, not actively monitored	• Monitoring metrics with predefined thresholds	• Advanced analytics for anomaly detection	• Automated scaling, anomaly detection based on ML
Tracing	• Tracing not implemented	• Basic tracing for critical services	• Distributed tracing for performance optimization	• Automated tracing, root cause analysis
Canaries	• Canaries not utilized	• Basic canaries for critical services	• Advanced canaries for predictive insights	• Self-adaptive canaries, automatic scaling
Real User Monitoring (RUM)	• RUM data not collected	• Basic RUM data collection for user experience	• Advanced analytics for predicting user behavior	• Automated optimization based on RUM and analytics

AWS Pillars from Reactive to Autonomous (Cont.)

Pillars of AWS Observability	Reactive	Proactive	Predictive	Autonomous
Infrastructure Monitoring	• Basic metrics collected, not actively monitored	• Automated monitoring with alerts for deviations	• Predictive maintenance and capacity planning	• Self-healing infrastructure, automated scaling
Network Monitoring	• Network monitoring tools not implemented	• Basic network monitoring for outages/performance	• Advanced analytics for security threats	• Self-adaptive network monitoring, dynamic config
Security Monitoring	• Security monitoring not implemented	• Basic monitoring tools for known threats	• Real-time threat detection, automated response	• Autonomous security monitoring with AI
Cost Optimization	• Cost optimization not considered	• Basic strategies based on manual analysis	• Advanced optimization using automation/predictive	• Fully automated cost optimization, CI/CD integrated

Measure Progress with Business Outcomes

- **Mean Time to Detect (MTTD):** Decrease the time it takes to identify issues.
- **Mean Time to Resolve (MTTR)**: Shorten the time it takes to detect and fix issues.
- Mean Time Between Failures (MTBF): Increase the interval between system failures.
- Improved System Reliability and Availability: Enhance system uptime and minimize downtime.
- Enhanced User Experience: Boost user satisfaction with faster and smoother interactions.
- Optimized Resource Utilization: Ensure efficient use of computing resources to save costs.
- Increased Development Velocity: Accelerate the delivery of new features and updates.
- Alignment with Service Level Objectives (SLOs): Ensure observability efforts meet defined performance targets and business objectives.

Best practices

- **Standardize Logging and Monitoring**: Use AWS CloudWatch Logs and Metrics for centralized logging and monitoring. Maintain consistency in log formats and naming conventions.
- **Instrumentation with AWS X-Ray**: Implement distributed tracing with X-Ray for visibility into request flows across microservices and AWS resources. Use X-Ray SDKs for seamless integration and performance analysis.
- **Automated Alerting and Response**: Configure CloudWatch Alarms to trigger alerts based on predefined thresholds. Set up automated response actions with Lambda or SNS for quick issue resolution.
- **Continuous Performance Optimization**: Utilize AWS Compute Optimizer to analyze resource utilization and recommend optimal configurations. Monitor and optimize infrastructure based on performance metrics and usage patterns.
- **Integration with AWS Managed Services**: Leverage managed services like RDS, DynamoDB, and Lambda for enhanced observability. Utilize built-in monitoring features and CloudWatch integrations for insights into service performance.

Thank you.