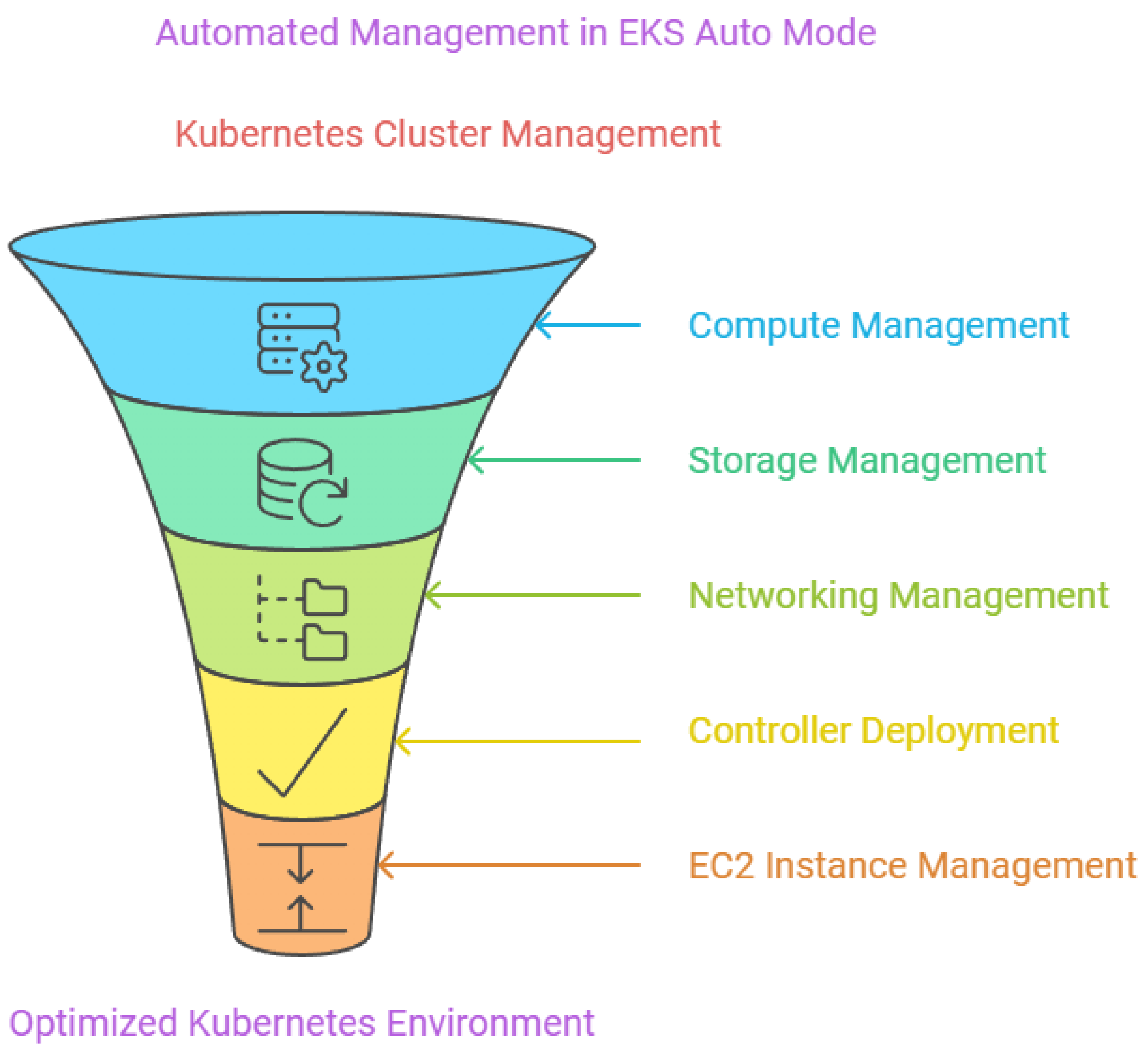
# Introduction to Amazon EKS Auto Mode

Mohan Ramadoss

## What Is Amazon EKS Auto Mode?



# Key Features of EKS Automatic Mode

EKS Auto Mode

Dynamic Scaling

Adjusts resources in real-time based on application demands.

Cost Optimization

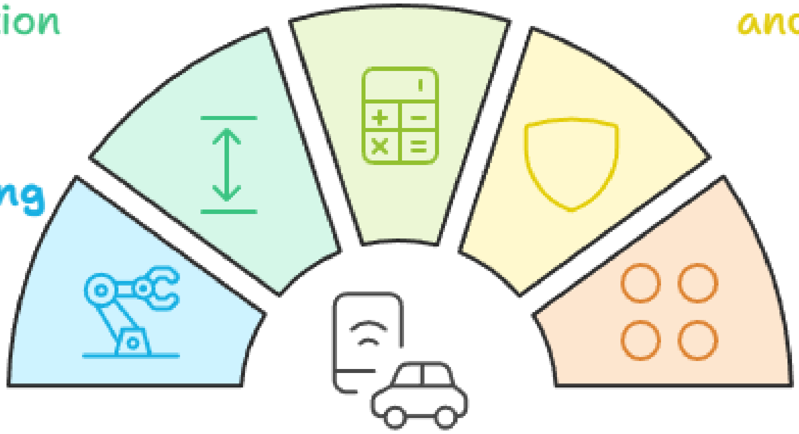
Continuously optimizes compute resources to reduce costs.

Security and Updates

Manages OS patching and security updates automatically.

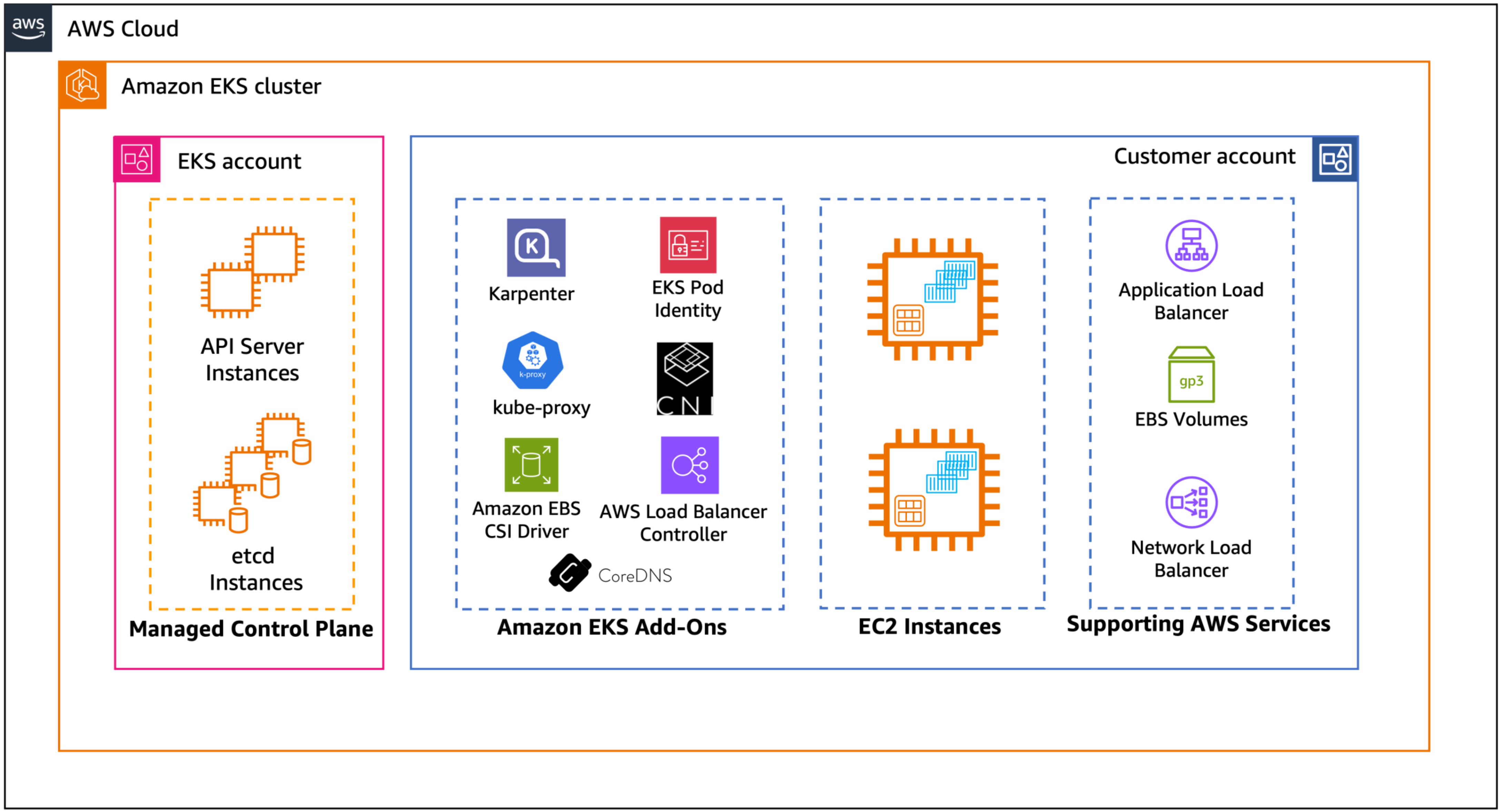
**Automated Provisioning**

Automatically provisions essential resources like EC2 instances and EBS volumes.

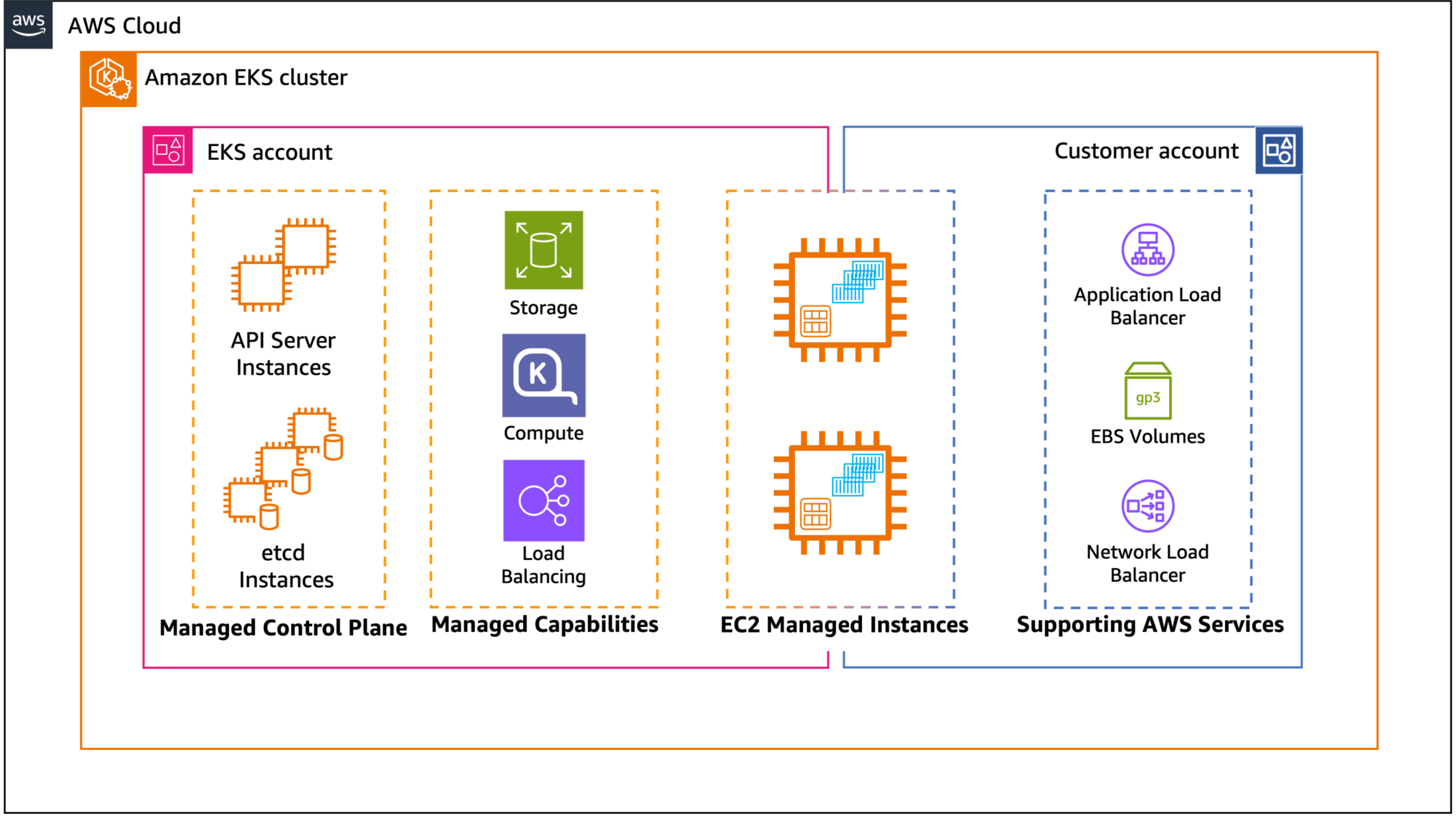


Simplified Management

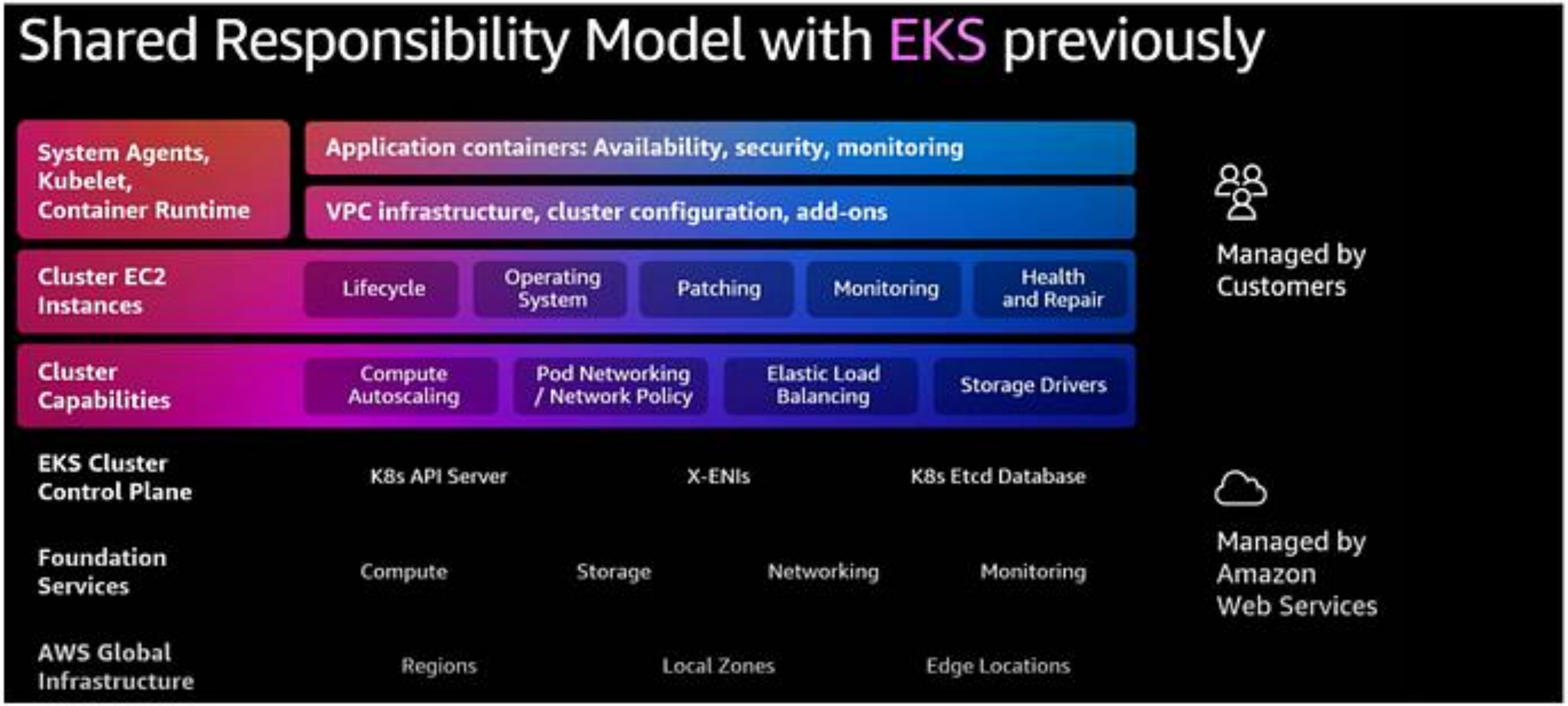
Reduces operational overhead by adhering to AWS best practices.



*Before Auto Mode*

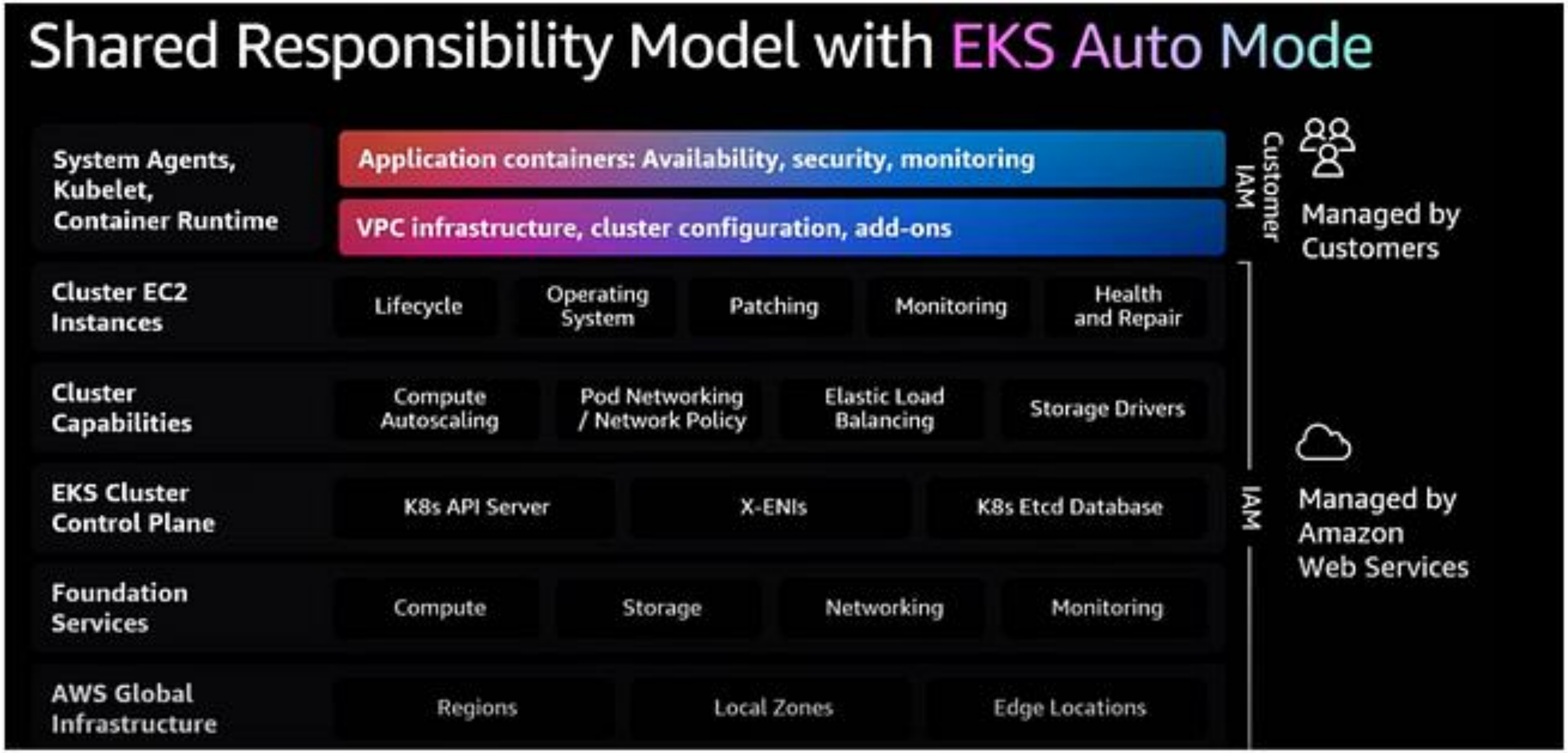


*After Auto Mode*



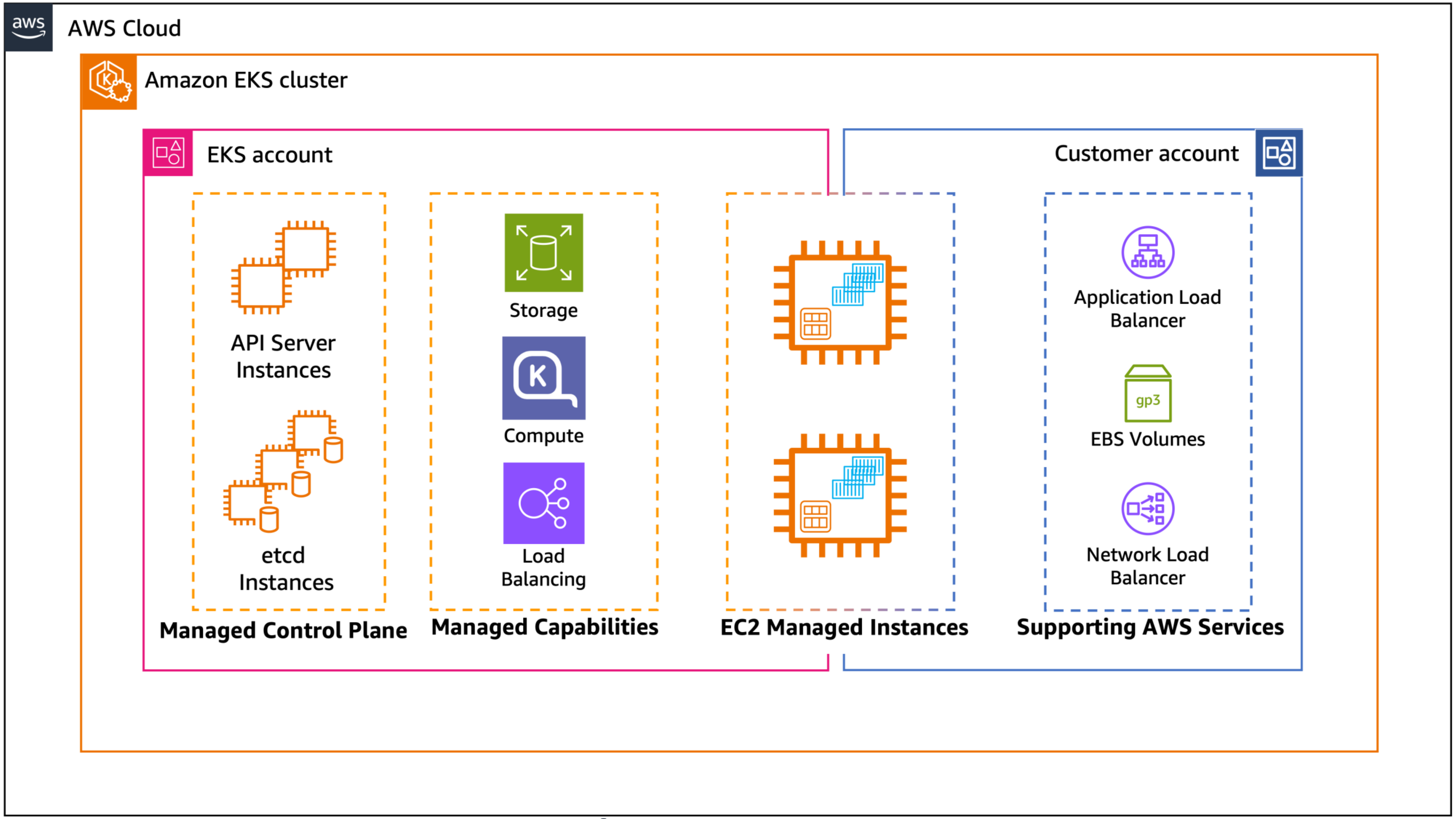
## Shared Responsibility Model with EKS previously

| | | | | | |
|---|---|---|---|---|---|
| System Agents, Kubelet, Container Runtime | Application containers: Availability, security, monitoring | | | | Managed by Customers |
| | VPC infrastructure, cluster configuration, add-ons | | | | |
| Cluster EC2 Instances | Lifecycle | Operating System | Patching | Monitoring | Health and Repair |
| Cluster Capabilities | Compute Autoscaling | Pod Networking / Network Policy | Elastic Load Balancing | Storage Drivers | |
| EKS Cluster Control Plane | K8s API Server | X-ENIs | K8s Etcd Database | | Managed by Amazon Web Services |
| Foundation Services | Compute | Storage | Networking | Monitoring | |
| AWS Global Infrastructure | Regions | Local Zones | Edge Locations | | |



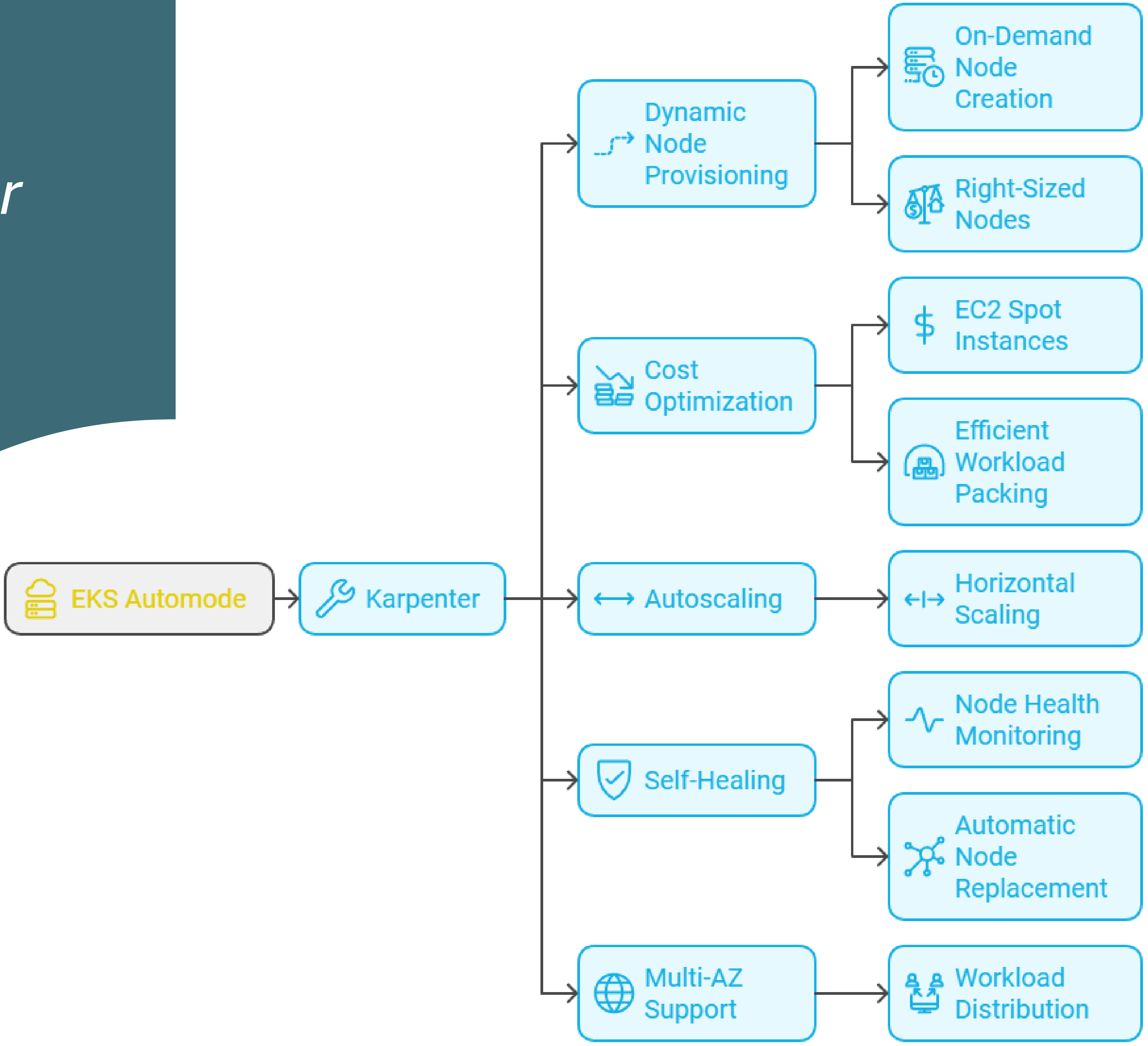
## Shared Responsibility Model with EKS Auto Mode

| | | | | | | |
|---|---|---|---|---|---|---|
| System Agents, Kubelet, Container Runtime | Application containers: Availability, security, monitoring<br>VPC infrastructure, cluster configuration, add-ons | | | | | Customer IAM — Managed by Customers |
| Cluster EC2 Instances | Lifecycle | Operating System | Patching | Monitoring | Health and Repair | IAM — Managed by Amazon Web Services |
| Cluster Capabilities | Compute Autoscaling | Pod Networking / Network Policy | Elastic Load Balancing | Storage Drivers | | |
| EKS Cluster Control Plane | K8s API Server | X-ENIs | K8s Etcd Database | | | |
| Foundation Services | Compute | Storage | Networking | Monitoring | | |
| AWS Global Infrastructure | Regions | Local Zones | Edge Locations | | | |

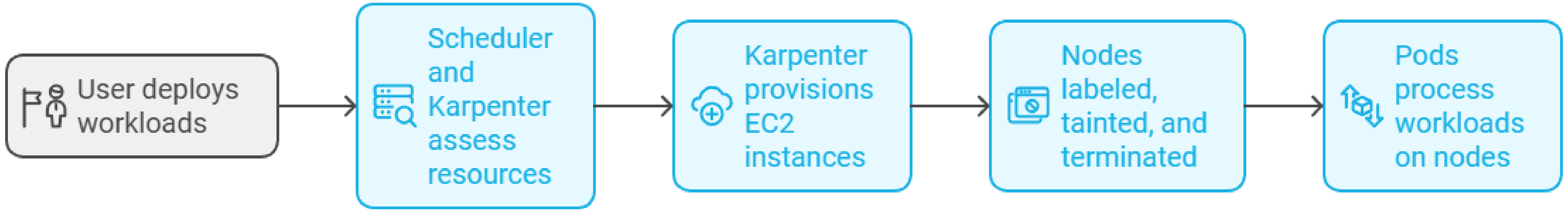


*Architecture overview*

# Karpenter

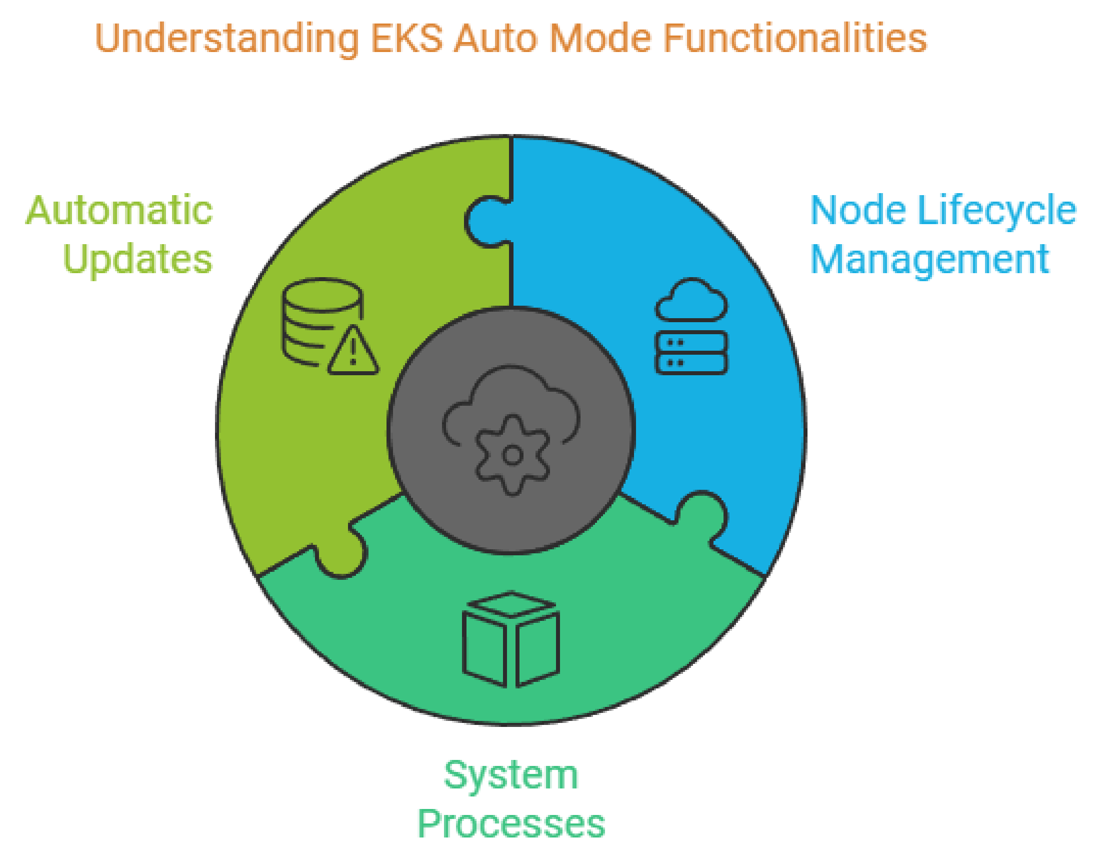

EKS Automode
Karpenter
Dynamic Node Provisioning
On-Demand Node Creation
Right-Sized Nodes
Cost Optimization
EC2 Spot Instances
Efficient Workload Packing
Autoscaling
Horizontal Scaling
Self-Healing
Node Health Monitoring
Automatic Node Replacement
Multi-AZ Support
Workload Distribution

# Workflow of EKS Auto Mode with Karpenter


User deploys workloads
Scheduler and Karpenter assess resources
Karpenter provisions EC2 instances
Nodes labeled, tainted, and terminated
Pods process workloads on nodes

## Key Functionalities of EKS Auto Mode

- **Node Lifecycle Management:** Manages the lifecycle of EC2 instances, including scaling and replacement of unhealthy nodes.
- **System Processes:** Runs node capabilities as system processes managed by AWS, not as Kubernetes DaemonSets.
- **Automatic Updates:** Handles cluster upgrades and OS updates automatically.

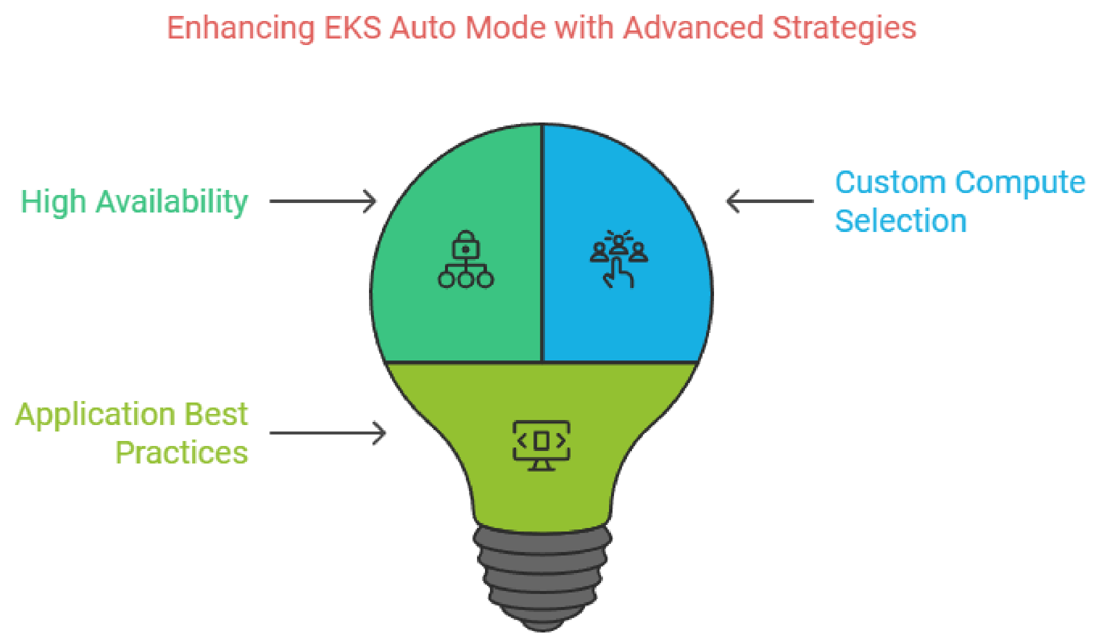

Understanding EKS Auto Mode Functionalities
Automatic Updates
Node Lifecycle Management
System Processes

# Advanced Use Cases and Configurations

- **High Availability:** Configure pod topology spread constraints for high availability across nodes and zones.

- **Custom Compute Selection:** Use Kubernetes labels to specify compute requirements for workloads.

- **Application Best Practices:** Implement pod disruption budgets, resource requests/limits, and graceful shutdowns.


Enhancing EKS Auto Mode with Advanced Strategies
High Availability
Custom Compute Selection
Application Best Practices

| Method | Description | Use Case |
|---|---|---|
| AWS Management Console | Simple graphical interface to configure and create the cluster with Auto Mode in a few clicks. | Quick setups and minimal technical expertise. |
| AWS CLI | Command-line tool to script and automate the creation of Auto Mode clusters. | Scripted workflows and automation. |
| Terraform | Infrastructure as Code (IaC) tool for declarative provisioning of EKS clusters. | Scalable and repeatable setups. |
| AWS CloudFormation | Service to define and deploy infrastructure templates, including EKS Auto Mode clusters. | Integrations with broader AWS setups. |
| CDK (AWS Cloud Development Kit) | Use programming languages like Python, JavaScript, or TypeScript to define cluster infrastructure. | Developers preferring code-driven approaches. |

# DEMO

- **Deploy a sample inflate workload to an Amazon EKS Auto Mode cluster**

Shows how to deploy a sample workload to an EKS Auto Mode cluster using kubectl commands.

- **Deploy a sample load balancer workload to EKS Auto Mode**

Shows how to deploy a containerized version of the 2048 game on Amazon EKS.

- **Deploy a sample stateful workload to EKS Auto Mode**

Shows how to deploy a sample stateful application to an EKS Auto Mode cluster.

# Feature Comparison

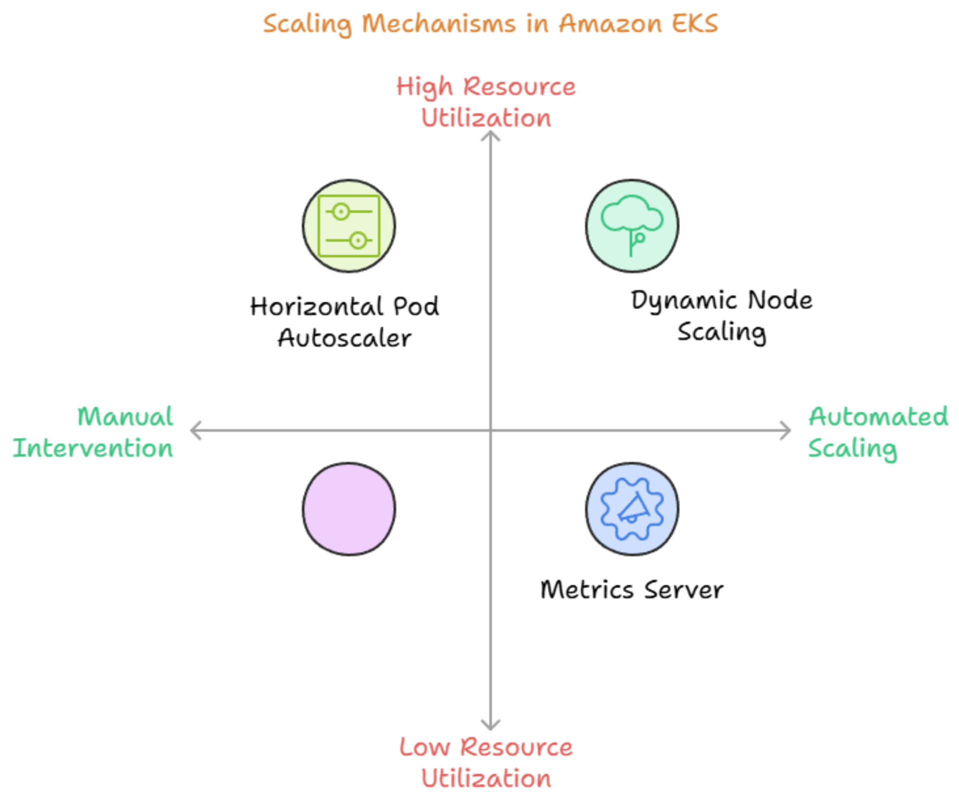
| Feature | EKS Auto Mode | EKS Standard Mode |
|---|---|---|
| **Node Management** | **Fully managed** - AWS manages the lifecycle of EC2 instances, including scaling, replacement of unhealthy nodes, and automatic updates. | **User-managed** - Users are responsible for managing EC2 instances, including scaling, updates, and node health. |
| **Scaling** | **Automatic, optimized** - Scales resources dynamically based on workload demands, ensuring efficient resource utilization. | **Customizable** - Users can configure scaling policies but must manage scaling themselves. |
| **Target Audience** | **Beginners, teams wanting simplicity** - Ideal for those who want to focus on application development rather than infrastructure management. | **Advanced Kubernetes users** - Suited for users who need fine-grained control over their Kubernetes environment. |
| **Use Cases** | **Quick setups, auto-scaling** - Perfect for development environments, batch processing, and containerized applications with fluctuating resource needs. | **Fine-tuned, custom setups** - Suitable for complex workloads requiring custom networking, advanced configurations, or specific compute requirements. |
| **Operational Complexity** | **Reduced** - Automates Kubernetes setup and infrastructure scaling, making it beginner-friendly. | **Higher** - Requires in-depth knowledge of Kubernetes and AWS services for optimal management. |
| **Cost Efficiency** | **Optimized** - Eliminates over-provisioning by scaling compute resources dynamically based on actual workload needs. | **Manual optimization** - Users must manually manage and optimize resource usage to control costs. |
| **Security and Compliance** | **Improved** - Managed updates, patching, and integration with AWS security services enhance cluster security. | **User-managed** - Users must ensure security updates and compliance themselves. |
| **Flexibility** | **Limited** - Provides a streamlined experience with less flexibility for custom configurations. | **High** - Offers full control over cluster configurations, networking, and compute selection. |

# Scaling Beyond Default Limits

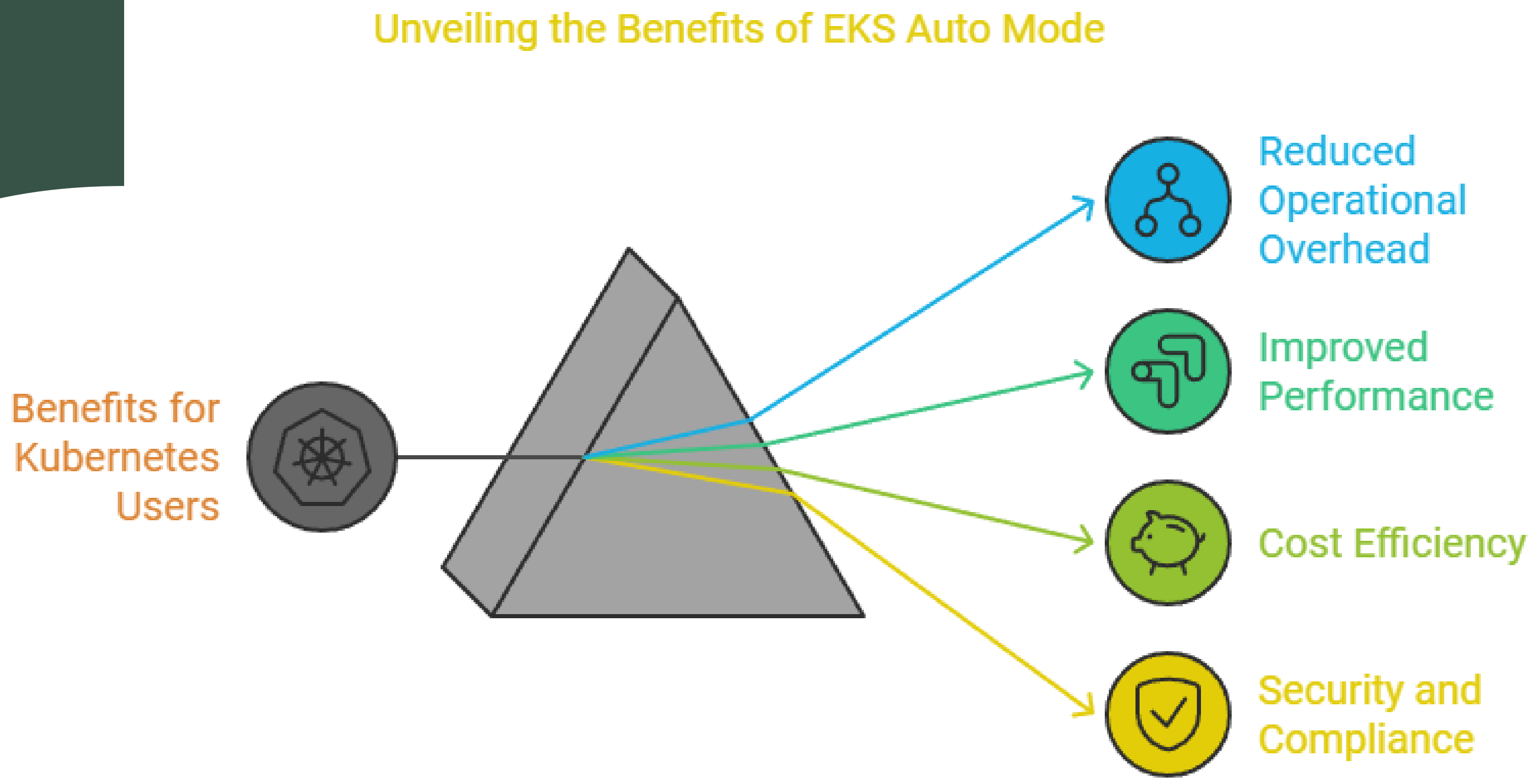
- **Horizontal Pod Autoscaler (HPA):** Automatically scales the number of pods based on CPU usage or other metrics.

- **Metrics Server:** Collects resource usage data for HPA to make scaling decisions.

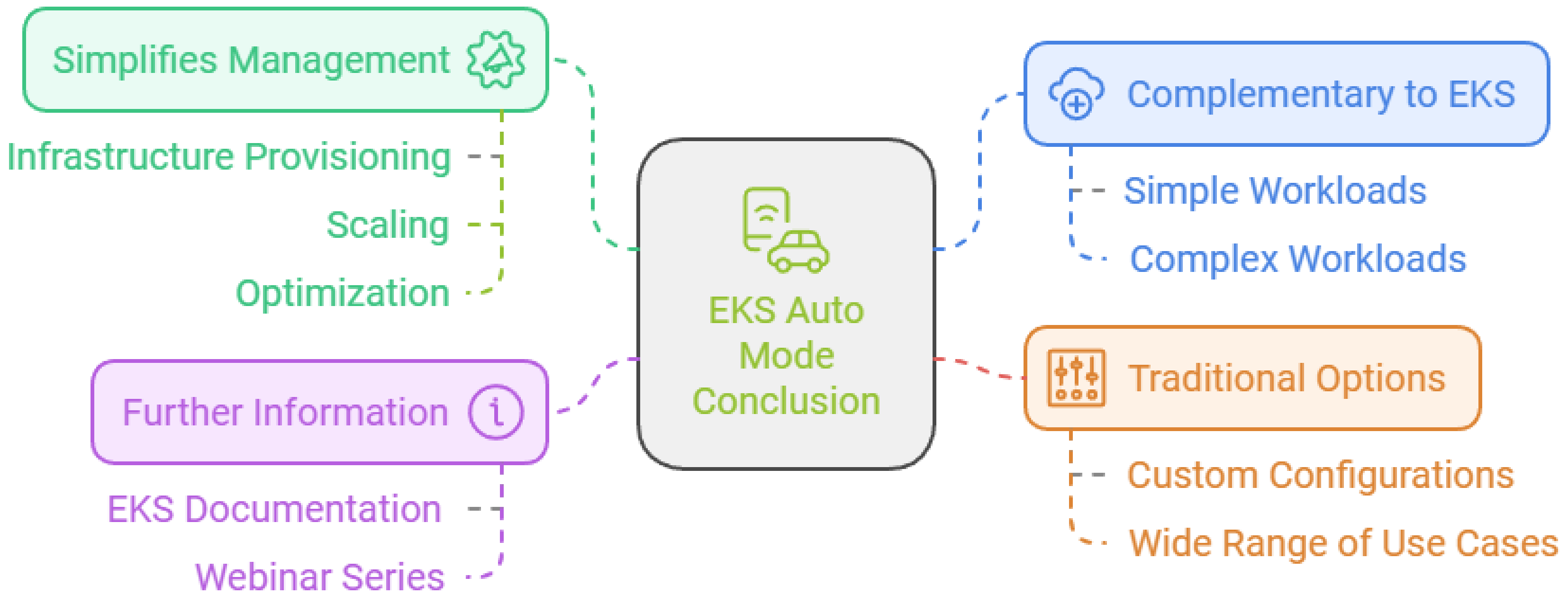
- **Dynamic Node Scaling:** EKS Auto Mode scales EC2 instances based on workload demands.



## Benefits for Kubernetes Users


Unveiling the Benefits of EKS Auto Mode
Benefits for Kubernetes Users
Reduced Operational Overhead
Improved Performance
Cost Efficiency
Security and Compliance

## Conclusion



Simplifies Management
Infrastructure Provisioning
Scaling
Optimization
Further Information
EKS Documentation
Webinar Series
EKS Auto Mode Conclusion
Complementary to EKS
Simple Workloads
Complex Workloads
Traditional Options
Custom Configurations
Wide Range of Use Cases