Kumomta Cloud Setup – POC Documentation

# 1. Introduction

Kumomta (Kumo Mail Transfer Agent) is a modern, cloud-native mail transfer agent designed to meet the needs of scalable, observable, and customizable email systems. It is built using async Rust and leverages Lua for scripting flexible mail routing logic. This document provides detailed guidance for setting up a Kumomta-based email infrastructure as part of a proof-of-concept (POC) on cloud environments.

# 2. What, Why, and Where

• What: Kumomta is an open-source MTA developed by Fastmail, intended as a cloud-native, high-performance replacement for traditional MTAs like Postfix and commercial tools like PMTA.  
• Why: To achieve better scalability, observability, and customization for cloud mail delivery.  
• Where: Suitable for deployment on Linux VMs, Docker containers, and Kubernetes clusters.

# 3. Key Features

• Async Rust engine with excellent concurrency  
• Lua-based scripting for message routing and filtering  
• Built-in observability with Prometheus metrics and structured logs  
• Cloud-native design with support for Docker and K8s  
• Hot-reloadable configuration  
• TLS encryption, SPF/DKIM/DMARC support

# 4. Pros and Cons

Pros:  
• Cloud-native and lightweight  
• High performance using async Rust  
• Excellent observability support  
• Extensible with Lua scripting  
• Free and open-source  
  
Cons:  
• Still under active development  
• Smaller community and less documentation  
• Lua scripting required for advanced logic  
• No GUI tools (CLI only)

# 5. Mail Sending Capacity

Kumomta has no hard-coded limit. On a 4-core, 8GB RAM cloud VM, it can handle over 1 million emails/day with proper tuning. Kubernetes scaling can support tens of millions of emails daily. Actual capacity depends on CPU, memory, IOPS, and network bandwidth.

# 6. Comparison with PMTA

Feature Comparison:  
• License: Kumomta (MIT, open-source), PMTA (commercial)  
• Config Format: Kumomta (YAML + Lua), PMTA (XML)  
• Observability: Kumomta (Prometheus, structured logs), PMTA (custom setup)  
• IP Warmup: Kumomta (scriptable), PMTA (built-in)  
• GUI: Kumomta (none), PMTA (has admin console)  
• Cost: Kumomta (free), PMTA (licensed per IP/domain)

# 7. Architecture Overview

Kumomta consists of a core Rust engine, configurable SMTP listeners, Lua scripting hooks, an asynchronous mail queue system, and built-in support for logging and metrics. The architecture is modular and highly scalable.

# 8. Installation Options

• Binary download (Linux)  
• Docker container (for cloud or dev)  
• Kubernetes deployment via Helm chart

# 9. Configuration Example

Sample YAML:  
server:  
 listen:  
 - addr: "0.0.0.0:25"  
 protocol: smtp  
 hostname: "mail.example.com"  
  
logging:  
 level: info  
 format: json  
  
metrics:  
 enabled: true  
 listen: "0.0.0.0:9090"  
  
routing:  
 script: "routing.lua"

# 10. Sample Lua Routing

require("kumomta")  
  
on\_connect(function(session)  
 session:accept()  
end)  
  
on\_mail\_from(function(session, from)  
 return ACCEPT  
end)  
  
on\_rcpt\_to(function(session, recipient)  
 return ACCEPT  
end)  
  
on\_data(function(session, message)  
 return DELIVER  
end)

# 11. POC Checklist

• [ ] Deploy Kumomta on VM/Docker/K8s  
• [ ] Test sending/receiving emails  
• [ ] Implement Lua-based routing  
• [ ] Setup Prometheus for metrics  
• [ ] Configure SPF/DKIM/DMARC records  
• [ ] Integrate logs with observability stack

# 12. Resources

• GitHub: https://github.com/KumoCorp/kumomta  
• Docs: https://docs.kumomta.com  
• Video: https://www.youtube.com/watch?v=vA8h6cQZvjg