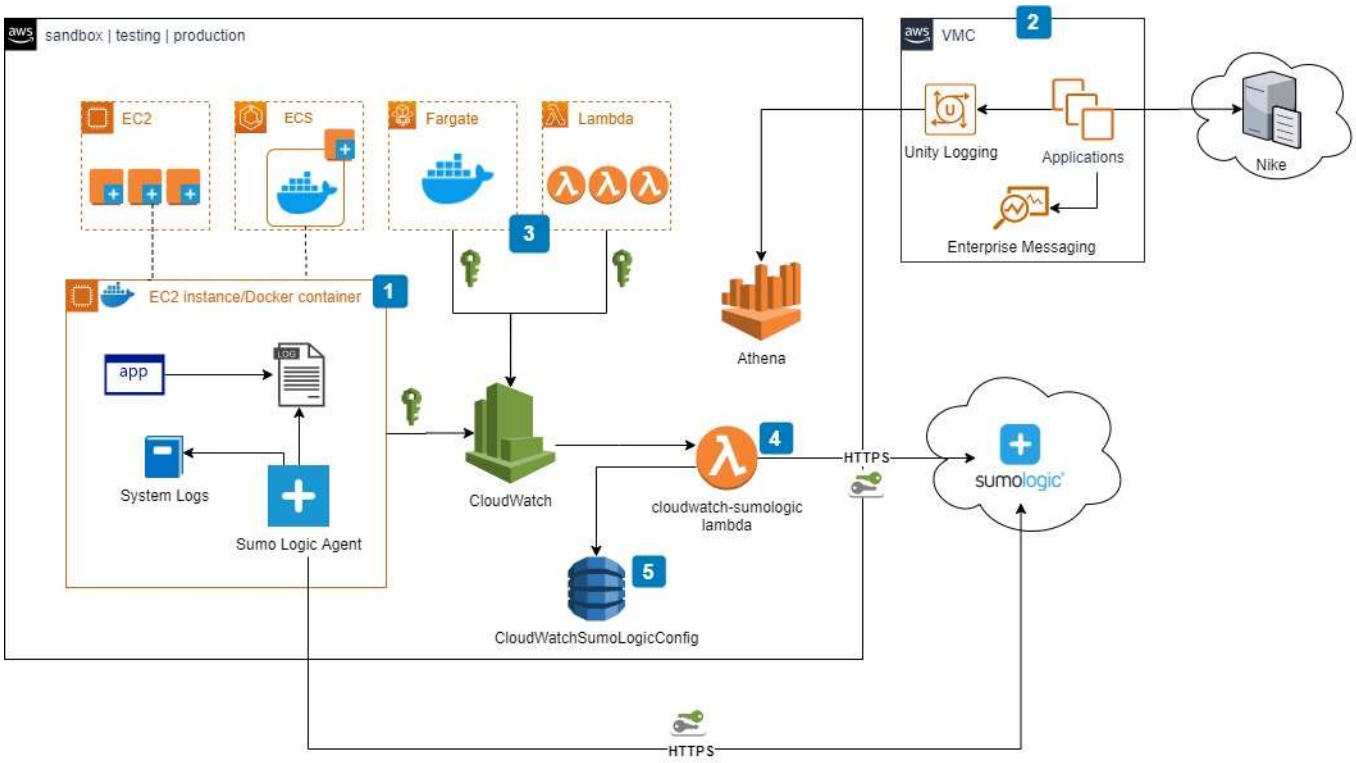
# Logging

This article discusses current application logging practices for applications running with AWS native services. Some of these practices have been deprecated in favor of others to provide a cohesive, flexible arechitecture for application logging.

## Current State

This is the current landscape of application logging, depicting the various ways and technologies used to generate logs.




1. EC2 instance logging is very similar to Docker containers in ECS. Some information goes to CloudWatch, while most logs go directly to Sumo Logic via the installed collector.
2. Containers that are running in Fargate and Lambda functions send stdout/stderr directly to CloudWatch.
3. Integration with Sumo Logic or any external log provider is done through a Lambda function that will send logs over HTTPS.
4. Configuration for log groups to send to Sumo Logic are specified in this Dynamo table. Terraform module to configure a log group to be sent to Sumo Logic.

## Deprecated Logging Mechanisms

The following practices are considered deprecated and should not be used for greendfield applications. Existing applications that are migrated to AWS native should consider updating to the desired pattern in the following section.

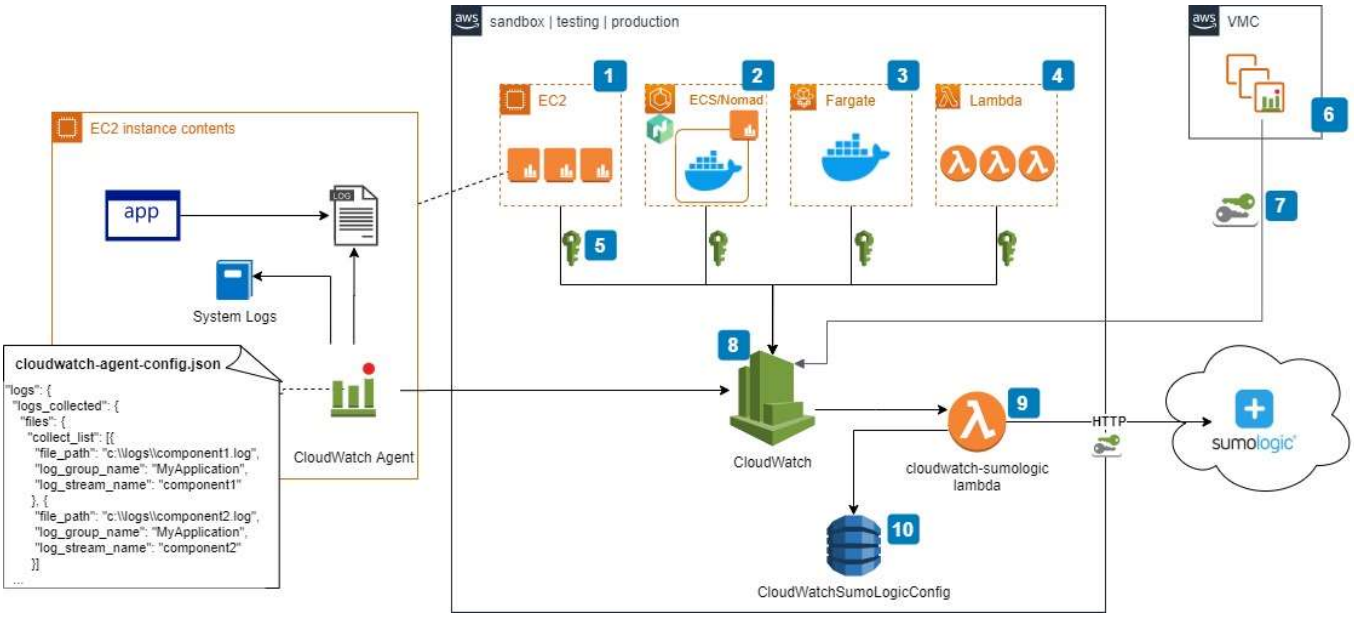
❌ Sumo Logic Agent
❌ Enterprise Messaging
❌ Log files to Nike
❌ Unity Logging

# Desired State

## Design Goals

The desired state is a reflection of the following design goals:

1. Decouple applications from the underlying log provider to avoid the need to change application code when logging providers change. This is accomplished by logging to a file or stdout/stderr instead of tying the application directly to a logging provider.
2. Provide a central point of integration with any external logging providers to manage configuration and provide flexibility to migrate to another logging provider (or cut it off) without needing to change deployed resources.
3. Provide a consistent logging pattern for applications running in AWS native.




1. EC2 instances use the CloudWatch agent to send system logs and contents of log files. The agent does not support capturing stdout/stderr so applications should redirect stdout/stderr to a file and configure the agent to send those files to CloudWatch. Each application can

write to a unique log group and/or log stream. Documentation for CloudWatch agent configuration

2. ECS and Nomad clusters that are running EC2 instances use the CloudWatch agent to push desired system logs. Applications running in containers are configured to use the awslogs driver to send stdout/stderr directly to CloudWatch.
3. Containers that are running in Fargate are configured to use the awslogs driver to push stdout/stderr directly to CloudWatch. Documentation to configure ECS/Fargate tasks to use awslogs Docker driver
4. Lambda functions natively send stdout/stderr to CloudWatch when the function is configured to do so.
5. Resources in AWS will use IAM role authentication with an attached policy to allow writing logs to CloudWatch
6. CloudWatch is the central integration point for all logs
7. Integration with Sumo Logic or any external log provider is done through a Lambda function that will send logs over HTTP.
8. Configuration for log groups to send to Sumo Logic are specified in this Dynamo table. Terraform module to configure a log group to be sent to Sumo Logic.