Cost Optimization Tips for AWS IOT Workloads: A Deeper Dive into Payload Optimization with Kepware IOT Gateway and AWS IoT Core

Cost Optimization for AWS IOT Workloads is a key topic when dealing with high-volume IOT workloads in production. The AWS blog post "*Cost Optimization Tips for AWS IoT Workloads*" introduces valuable AWS tools designed to help manage IoT costs effectively. Building on those insights, this article offers practical strategies to enhance cost efficiency further, focusing on strategic adjustments to edge device configurations.

A key aspect highlighted in the AWS blog is optimizing message size and frequency to reduce cloud expenses. Here, we dive deeper into this topic to provide specific techniques to achieve these cost savings. Edge device configurations play a crucial role in workloads

streaming real-time data from edge devices to cloud platforms like AWS IoT Core or directly to services such as AWS Kinesis. Given the billing structures of these platforms, fine-tuning these settings can significantly impact your overall cloud costs.

- What additional benefits can be derived from fine-tuning your edge configurations for better cloud cost management?
- How should you proceed with such optimizations?

> ***If your message sizes are not significantly larger than the size that triggers an event then you might already be close to an optimal point***

In summary, check the distribution of your message sizes using tools such as AWS CloudWatch. If your message sizes are significantly larger than the size that triggers an event (e.g., 5 KB for AWS IoT Core), you might already be close to an optimal point. However, if the message sizes are not consistently significantly larger (~3x) than this threshold, there could be substantial savings from optimization efforts.

Edge devices like Kepware IoT Gateway offer configurable parameters like message buffering time. These settings directly influence the frequency of transmissions and the size of individual messages. However, finding the right balance isn't straightforward. Data from edge devices is often stochastic and unpredictable, leading to message size variability. Suboptimal configurations can lead to significantly increased costs.

To tackle this, we introduce the concept of **Billing Payload Efficiency (BPE)**—a metric that evaluates how well the transmitted payloads utilize the billable capacity for each event. Organizations can configure their edge systems by analyzing and optimizing this efficiency to achieve near-theoretical minimum costs while maintaining desired latency and reliability.

Introducing Billing Payload Efficiency

Billing Payload Efficiency (BPE) refers to how effectively the maximum capacity of a billable event is utilized. It can be calculated using the formula:

$$\textbf{Billing Payload Efficienicy} = \frac{\textbf{100} * \textbf{Payload Size (KB)}}{\textbf{No of Events for Payload Size} * \textbf{Event Capacity (KB)}}$$

This concept is illustrated below.

Message Size Range	Payload Size	Event Capacity	# of Events/ Payload Size	BPE	Average BPE
4 to 6	4.00	5	1	80.00	75.00
	4.25	5	1	85.00	
	4.50	5	1	90.00	
	4.75	5	1	95.00	
	5.00	5	1	100.00	
	5.25	5	2	52.50	
	5.50	5	2	55.00	
	5.75	5	2	57.50	
	6.00	5	2	60.00	

Message Size Range	Payload Size	Event Capacity	# of Events/ Payload Size	BPE	Average BPE
3 to 5	3.00	5	1	60.00	80.00
	3.25	5	1	65.00	
	3.50	5	1	70.00	
	3.75	5	1	75.00	
	4.00	5	1	80.00	
	4.25	5	1	85.00	
	4.50	5	1	90.00	
	4.75	5	1	95.00	
	5.00	5	1	100.00	

This implies that when the messages are between 4 and 6, assuming a uniform distribution as shown, the billing efficiency would be around 75%.

The illustration below shows how optimization through better message sizing and frequency can lead to a 26% reduction in costs.

	Message Characteristics				Standard Model		Optimized Model	
AWS Cost/ 1 Million Events	Min	Max	Max Event Size	Sum Payload/ Day	# of Events	Cost	# of Events	Cost
$1.00	3	5	5	40,000,000	10,000,000	$10.00	8,000,000	$8.00
$1.00	4	6	5	50,000,000	15,000,000	$15.00	10,000,000	$10.00
$1.00	6	12	5	90,000,000	23,333,333	$23.33	18,000,000	$18.00
					Average	$16.11		$12.00
							Cost Reduction Achieved through Optimization	26%

Stochastic Payload Distributions and Expected Billing Payload Efficiency

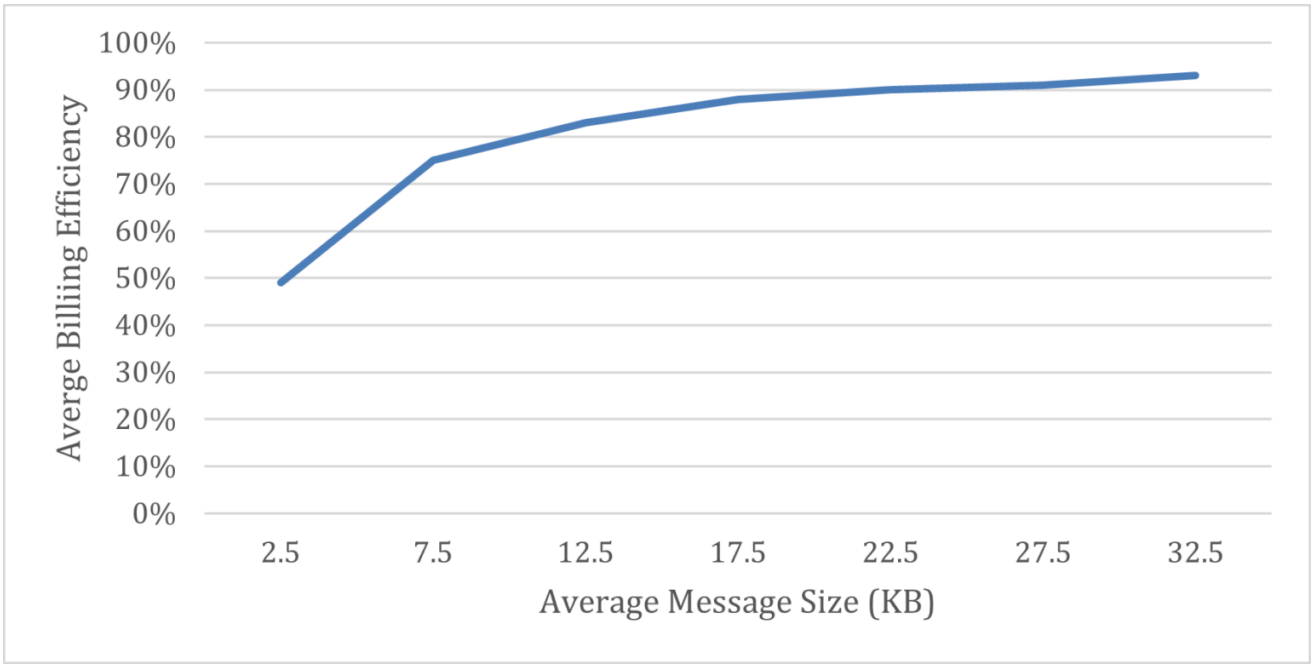
When working with edge devices like the **Kepware IoT Gateway**, the stochastic nature of data generation at the source introduces variability in payload sizes. For example, Kepware is often configured to publish data only when changes occur. Consider a sample scenario where a buffering interval of 1-second results in payload sizes varying significantly, ranging between 5 KB and 15 KB. Such randomness in payload sizes directly impacts billing efficiency and overall costs, making it crucial to optimize buffering strategies.

Introducing Average BPE

To account for this variability, we introduce the concept of Average BPE, which represents the expected value of BPE for a given payload size distribution. This metric provides a quantitative framework to evaluate and optimize buffering strategies under varying conditions.

Simulation and Analysis

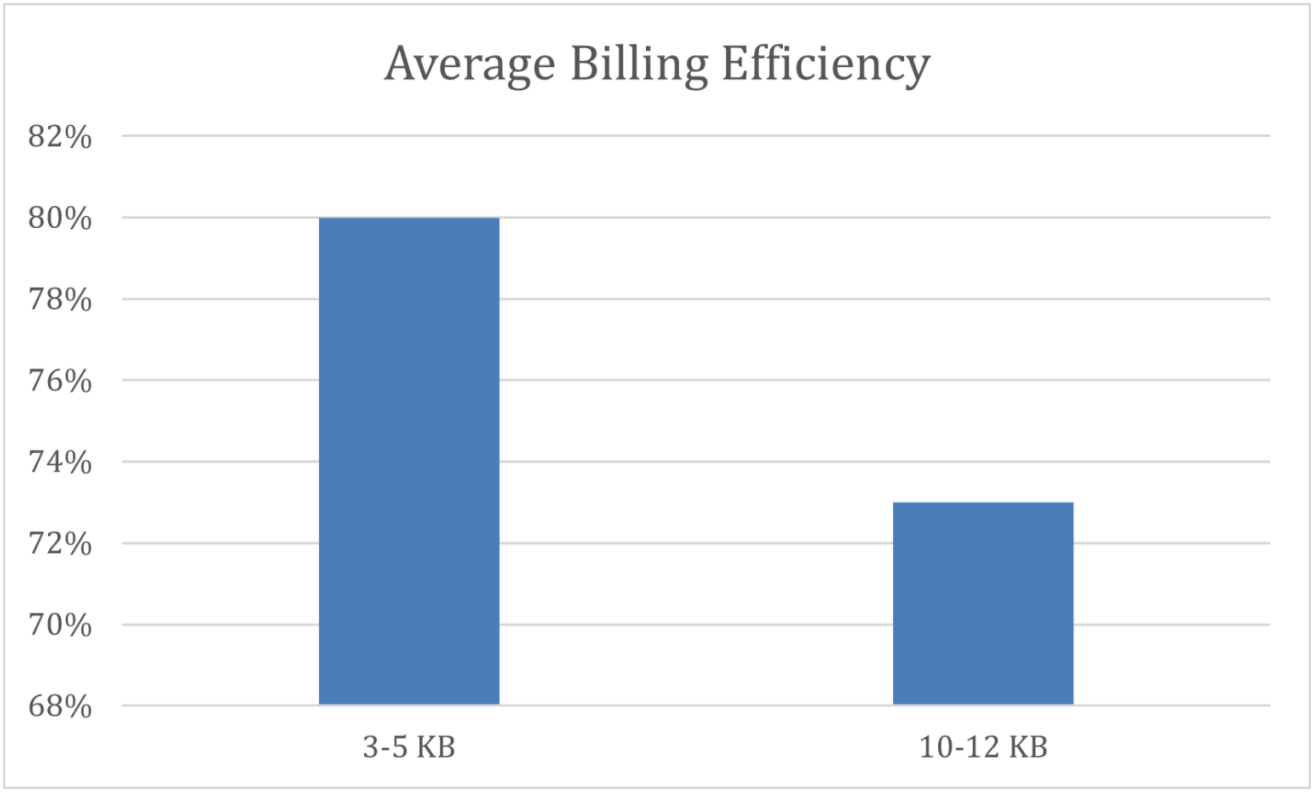
Average BPE can be estimated for various distributions of payload sizes. Assuming the payload is distributed uniformly with a range of 5KB, the average BPE versus the average message size is shown in the figure below. Although specific results vary depending on the distribution, the key principles remain consistent.



Let's analyze the above figure to understand the overall trend. When your average message size is 2.5 KB, you waste approximately 50% of the available billable event capacity. In other words, adopting a better buffering strategy could reduce your IoT Core costs by 50% for the same data volume.

Two key trends emerge from this analysis. First, smaller messages tend to have lower billing efficiencies due to how billable events are calculated. Second, billing efficiencies gradually converge toward 100% as message sizes increase.

The relationship is not a straightforward, monotonically decreasing one. For instance, consider the Average Billing Efficiency under two scenarios: in the first scenario, the payload varies uniformly between 3 and 5, resulting in an Average Billing Efficiency of 80%. In contrast, in the second scenario, where the payload varies uniformly between 10 and 12, the Average Billing Efficiency drops to 73%. As a result, it is essential to analyze the factors influencing efficiency across different payload ranges carefully.



While buffering for larger message sizes can significantly improve efficiency, it may not always be feasible due to latency constraints. The optimal strategy should balance cost efficiency and latency requirements tailored to your use case's needs and priorities.

It is important to note that the results depend on the actual payload distribution. To optimize for your payload, it is crucial to simulate the distribution accurately and then use tools to calculate the Average Billing Efficiency.

Many edge sources produce data at high volumes. Even at the highest transmission frequency, the payload size may be significant. For these scenarios, explicit optimization of payload size may only have marginal benefits since the number of billable events is already close to the optimal number. However, for scenarios where edges produce data at low volume, the right strategy can have a significant impact.

AWS IoT Core Billing Overview

AWS IoT Core charges based on **Message Size**: Messages up to 5 KB are considered a single billable event. Larger messages are split into multiple 5 KB chunks, each billed separately.

Payload Optimization

Benefits of Larger Messages

1. **Cost Savings**: Larger, efficiently sized messages reduce the number of billable events.
2. **Improved Downstream Processing**: Consolidated messages allow for bulk processing, enhancing system performance.

Trade-offs and Challenges in Optimizing Billing Payload Efficiency

1. **Increased Latency**: Higher buffering times can delay data transmission, impacting real-time processing requirements.
2. **Risk of Lost Messages**: Without store-and-forward capabilities, longer buffering times increase the impact of connectivity issues.

Advantages with DeepIQ Edge

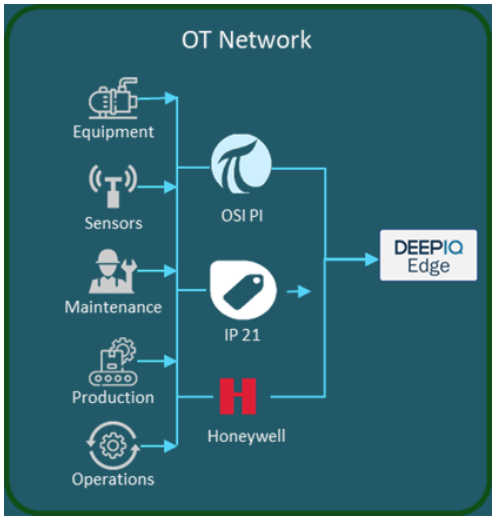


Figure1: DeepIQ Edge

DeepIQ Edge software is designed to enhance data transmission reliability and optimize cloud expenses for industrial applications. Here are its key advantages:

- Automated Payload Optimization: DeepIQ Edge employs advanced buffering techniques to ensure optimal transmission of messages. Adjusting buffer sizes and managing payloads efficiently maintains ideal sizes like 4.5 KB. This approach maximizes network efficiency and operational performance while minimizing cloud costs, especially in environments with variable data flow. Additionally, a timeout feature ensures timely data transmission, even if the buffer isn't full, which helps maintain consistency in message sizes and reliability.
- Support for Non-MQTT Payloads: Standard IOT workloads are built for AWS IOT Core messaging and Rules for forwarding the data to other AWS services. DeepIQ Edge can push the data directly into AWS services such as AWS S3 or Kinesis Data Streams,

eliminating the need for multiple hops and significantly reducing costs.

- Comprehensive Historian Compatibility: DeepIQ Edge is engineered to integrate seamlessly with a wide range of historians, including OSI PI, IP 21, and Honeywell PhD. This compatibility ensures efficient ingestion of historical data, which enhances the analytical capabilities and data utilization of industrial operations. By supporting the ingestion of historical data, DeepIQ Edge offers unmatched flexibility and utility in managing and analyzing industrial data streams.

These features position DeepIQ Edge as a robust edge solution for industrial settings, reducing operational costs while maintaining high data integrity and transmission efficiency

Optimization with Kepware

Kepware specializes in real-time connectivity to control systems and SCADA, enabling seamless communication with industrial devices for operational monitoring and control.

Configuring the ideal buffering time in Kepware IoT Gateway involves balancing latency, message size, and the potential impact of lost messages. Since Kepware IoT Gateway does not have store-and-forward capabilities, increasing the buffering time also increases the risk of losing more data during connectivity issues. If you have determined an ideal buffering time (e.g., 1500 ms), follow these steps to implement and monitor the configuration:

1. **Access Kepware Settings**:

- Log in to your Kepware server interface.
- Navigate to the IoT Gateway configuration.

2. **Configure the Buffering Time**:

- Select the specific IoT Gateway agent you want to configure.
- Locate the **Rate(ms)** setting.
- Set the Rate to your desired value (e.g., 1500 ms).
- Save the changes to apply the new buffering configuration.

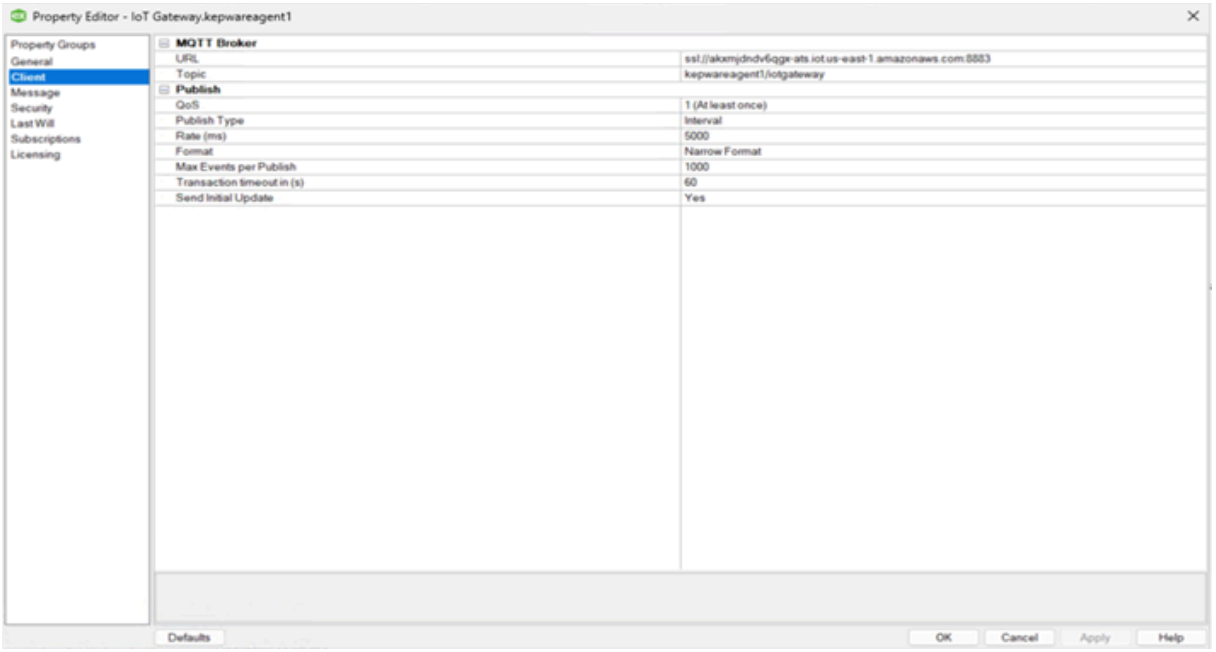


Figure 2: IoT Gateway

3. Monitor Logs in CloudWatch:

- Set up monitoring in AWS CloudWatch for IoT Core logs to track metrics such as the number of events, average message size, and latency
- Validate that the implemented buffering strategy aligns with cost and reliability goals.
- Track the distribution of message sizes and calculate average billing efficiency.

Figure 3: Monitor Logs

4. Adjust as Needed:

- Based on the CloudWatch metrics, fine-tune the buffering time in Kepware to optimize performance further.
- Ensure that any adjustments maintain a balance between cost efficiency and latency requirements.

Monitoring and Feedback Loop

- **Billing Reports**: Regularly review AWS IoT Core billing reports to measure cost impacts.
- **Kepware Logs**: Analyze logs to track message sizes and adjust configurations as needed.
- **Payload Efficiency Metrics**: Calculate Billing Payload Efficiency to identify inefficiencies and opportunities for improvement.

Conclusion

Billing Payload Efficiency is a new metric focused on IoT cost optimization. By aligning edge software configurations like DeepIQ Edge or Kepware IoT Gateway with Cloud platforms such as AWS IoT Core's billing model, organizations can significantly reduce costs while improving efficiency. The key is to buffer data intelligently, balancing payload size against transmission frequency and minimizing wasted bandwidth.

Optimizing edge payloads for cloud cost efficiency is vital in the broader context of IT-OT convergence. This strategy is just one piece of a complex puzzle that includes managing the lifecycle of edge software, contextualizing IT and OT data, developing robust cloud data models with stringent versioning and governance, and simplifying the implementation of AI and digital twin workflows. Each of these elements presents challenges and nuances that must be expertly navigated to unlock the full potential of digital transformation initiatives.

The DeepIQ platform has many capabilities and tools tailored to streamline complex IT-OT convergence tasks. It supports a comprehensive array of functions, from constructing edge asset hierarchies to advanced IT contextualization, and facilitates streaming AI and digital twin workflows. As a unified solution, DeepIQ simplifies integration and accelerates the deployment of digital strategies. Engineered to address the intricacies of these processes, our platform ensures meticulous management of every convergence aspect, perfectly aligning with your business objectives.

For organizations looking to explore the full spectrum of benefits that IT-OT convergence can offer, the DeepIQ platform is a proven

existing industrial automation technology stack.

Privacy Policy Terms of Services

FEATURES

- Extract
- Engineer
- Explore

INDUSTRIES

- Upstream
- Midstream
- Downstream & Chemical
- Mining

SOLUTIONS

- IT-OT Contextualization
- Well Construction Optimization
- P & ID Digitization
- Predictive Maintenance
- Production Optimization
- Route Optimization
- Sustainability

PARTNERS

- AWS
- Azure
- Cloudera
- Google
- Databricks
- OSIsoft(AVEVA)
- Snowflake

COMPANY

- About Us
- Resources
- News
- DataStudio Deployment Guide
- Career
- Contact Us

+1 (832) 391-6834 info@deepiq.com