Apache Kafka is designed to be highly performant, reliable, scalable, and fault tolerant. At the same time, the performance and reliability of a Kafka cluster is highly dependent on the underlying infrastructure. That interdependence makes the right infrastructure choices critical to any successful deployment. For users who have made the decision to deploy Kafka on the AWS Cloud (https://www.confluent.io/blog/design-and-deployment-considerations-for-deploying-apache-kafka-on-aws/), making the right choices on storage infrastructure can seem daunting. The reality is that selecting reasonable infrastructure is easier than you think.

Let's start by thinking about the Kafka cluster at a high level. At its core, the Kafka cluster is a set of servers that offer a shared service where data can be published and retrieved by external clients. Each server is referred to as a Kafka broker, and the data managed by the brokers is logically divided into distinct topics. Data for each topic is persisted locally on the brokers, in a replicated and partitioned manner that prevents data loss or catastrophic disruption if a broker fails. By design, Kafka clusters will automatically re-replicate data and re-balance the client connections when a broker node is lost from the cluster. The brokers are optimized to aggregate the physical I/O for the topic data, resulting in a general pattern of sequential operations against the storage tier. Readers interested in a more comprehensive discussion of the Kafka architecture can refer to the documentation (http://docs.confluent.io/current/kafka/operations.html#kafka-operations).

Consider what this implies for the underlying storage infrastructure in a Kafka Cluster. Obviously, the absolute performance is critically important… as higher performance reduces the time needed to persist the data as it arrives in the cluster as well as the time needed to retrieve data for a consume or a new cluster node when re-replication is needed. EBS volumes in AWS (https://www.confluent.io/blog/design-and-deployment-considerations-for-deploying-apache-kafka-on-aws/) are an excellent option here. They provide consistent levels of I/O performance (IOPS) and ultimate flexibility in their deployment. A properly designed Kafka cluster based on EBS storage can virtually eliminate the re-replication overhead that would be triggered by an instance failure, as the EBS volumes can be

reassigned to a new instance quickly and easily. And from an operations perspective, a Kafka cluster deployed against EBS storage can be shut down cleanly without risk of data loss, a capability not possible when using EC2 Local Instance Storage.

This is why we view the new st1 and sc1 EBS offerings from Amazon (http://phx.corporate-ir.net/phoenix.zhtml?c=176060&p=irol-newsArticle&ID=2158367) as very promising. At a cost up to 50% lower than earlier EBS offerings, and optimized for sequential I/O workloads, we observed that these storage volumes delivered the performance and reliability needed for Kafka environments. We will conduct more detailed testing and welcome hearing about what others have found. (See Amazon blog: EBS Update – New Cold Storage and Throughput Options (https://aws.amazon.com/blogs/aws/amazon-ebs-update-new-cold-storage-and-throughput-options/)) .

The other infrastructure components (CPU, memory, networking) also play an important role in the capabilities of any Kafka cluster. In future blogs, I'll discuss the considerations for those sub-systems in greater detail. It was important to start with storage, because reliable, persistent data platforms such as Kafka are impossible without it.