Windows Azure Storage – A Highly Available Cloud Storage Service with Strong Consistency

Brad Calder, Ju Wang, Aaron Ogus, Niranjan Nilakantan, Arild Skjolsvold, Sam McKelvie, Yikang Xu, Shashwat Srivastav, Jiesheng Wu, Huseyin Simitci, Jaidev Haridas, Chakravarthy Uddaraju, Hemal Khatri, Andrew Edwards, Vaman Bedekar, Shane Mainali, Rafay Abbasi, Arpit Agarwal, Mian Fahim ul Haq, Muhammad Ikram ul Haq, Deepali Bhardwaj, Sowmya Dayanand, Anitha Adusumilli, Marvin McNett, Sriram Sankaran, Kavitha Manivannan, Leonidas Rigas

Microsoft

Some of the slides were taken from Brad Calder presentation at 23rd ACM Symposium on Operating Systems Principles (SOSP).

http://blogs.msdn.com/b/windowsazure/archive/2011/11/21/windows-azure-storage-a-highly-available-cloud-storage-service-with-strong-consistency.aspx

1. Introduction
2. Global Partitioned Namespace
3. High Level Architecture
4. Stream Layer
5. Partition Layer
6. Application Throughput
7. Workload Profiles

Windows Azure Storage

- Scalable cloud storage
- In production since November 2008
- Strong consistency
- Global and scalable namespace/storage
- Disaster recovery

Windows Azure Storage Data Abstraction

- Blobs - File system in the cloud
- Tables - Massively scalable structured storage
- Queues - Reliable storage and delivery of messages

Global Partitioned Namespace

http(s)://**AccountName**.<service>.core.windows.net/**PartitionName/ObjectName**

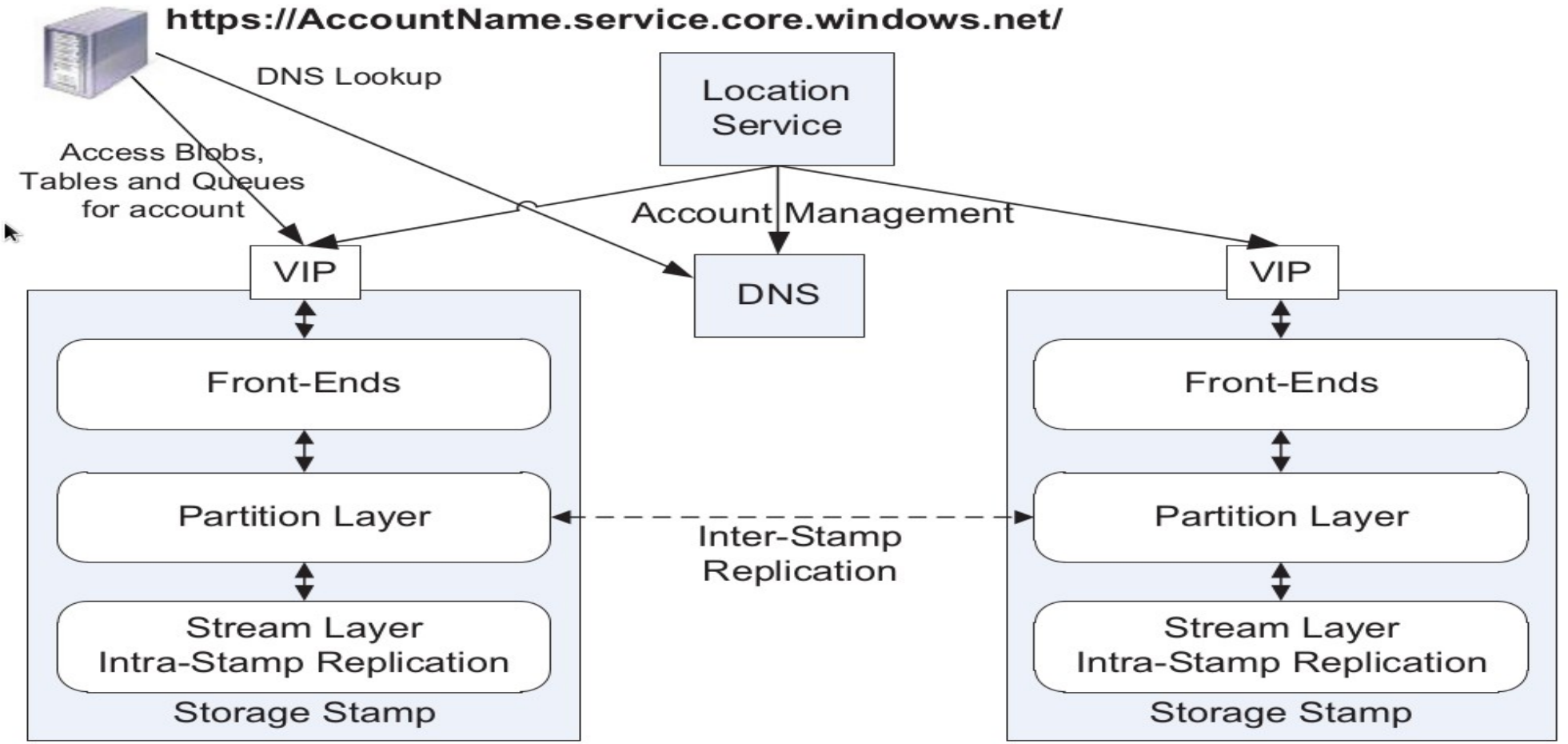
-<service> specifies the service type, which can be **blob, table,** or **queue**

High Level Architecture

Design Goals

- Highly Available with Strong Consistency
  - Provide access to data in face of failures/partitioning
- Durability
  - Replicate data several times within and across data centers
- Scalability
  - Need to scale to exabytes and beyond
  - Provide a global namespace to access data around the world
  - Automatically load balance data to meet peak traffic demands

High Level Architecture



Storage Stamp

- Cluster of 10 to 20 racks of starage nodes
- Each rack is built out as a seperate fault domain
- 18 disk-heavy storage nodes per rack
- 70% utilized in terms of capacity, transaction and bandwidth

Stream Layer

- Append-only distributed file system
- All data from the Partition Layer is stored into files(extents consisting of blocks) in the Stream Layer
- Each extent is repliacated 3 times(Intra-Stamp Replication)
- Does not understand higher level object(blob, table, queue)

Partition Layer

- Manages and understands high level data abstraction
- Uses Stream Layer interface to read and store objects in Stream Layer.
- Provides Inter-Stamp Repliaction
- Provides scalability by partitioning all of the data objects within a stamp

Front-End layer

- Consists of a set of stateless servers
- Authenticates and authorizes the request
- Routes the request to a partition server in the partition layer

Location Service

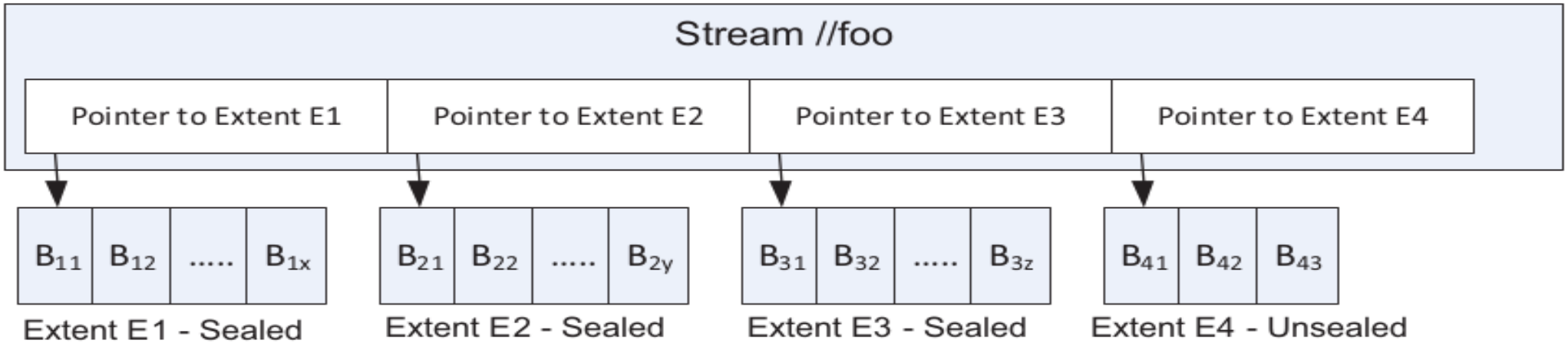
- Manages all the storage stamps
- Allocates accounts to storage stamps
- Distributed across two geographic locations for its own disaster recovery
- Ability to add new storage stamps

Stream Layer

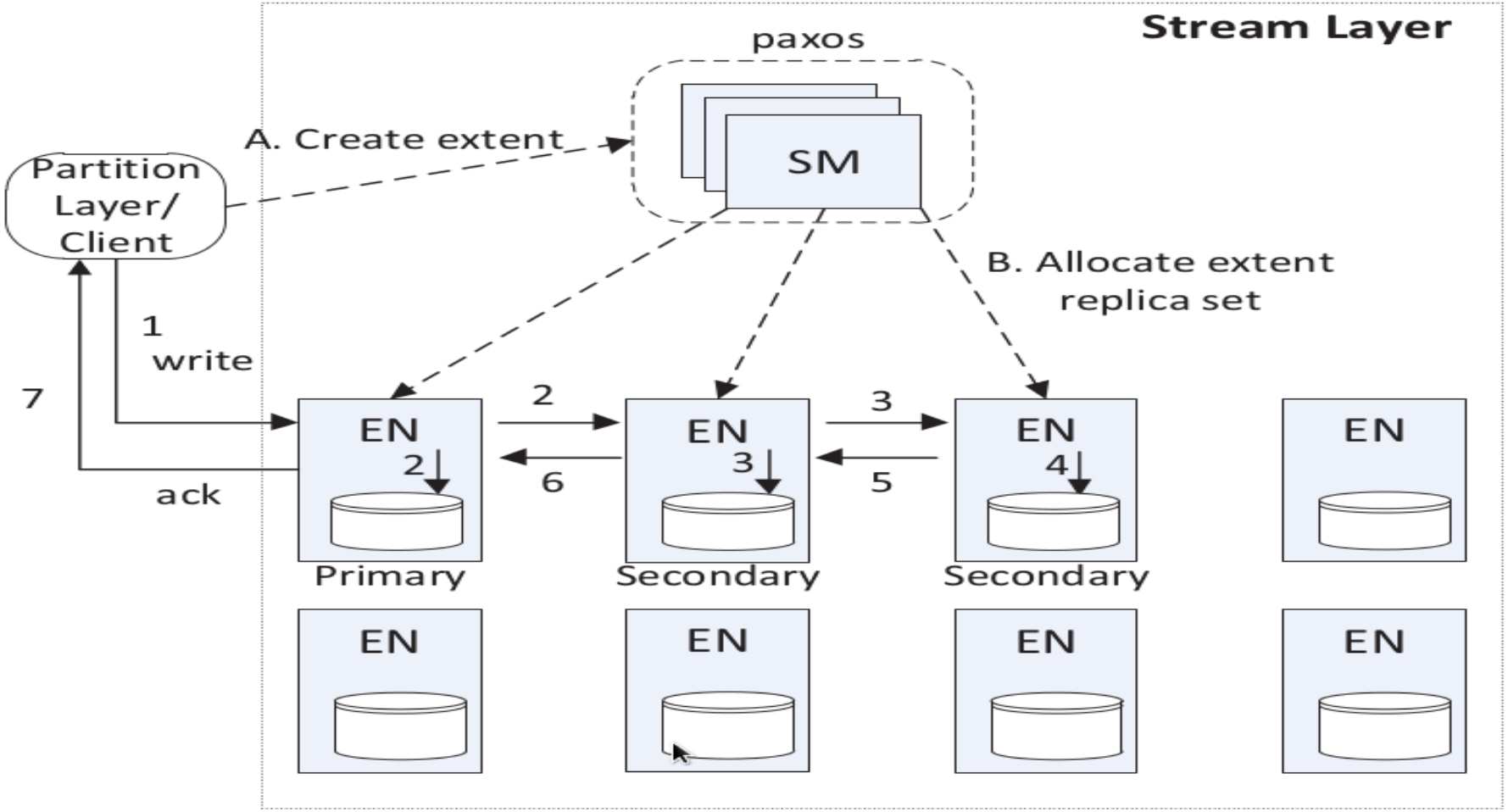
Stream Layer

- Append-Only Distributed File System
- Streams are very large files
  - Has file system like directory namespace
- Stream Operations
  - Open, Close, Delete Streams
  - Rename Streams
  - Concatenate Streams together
  - Append for writing
  - Random reads

Stream Layer Concept



Stream Manager and Extent Nodes



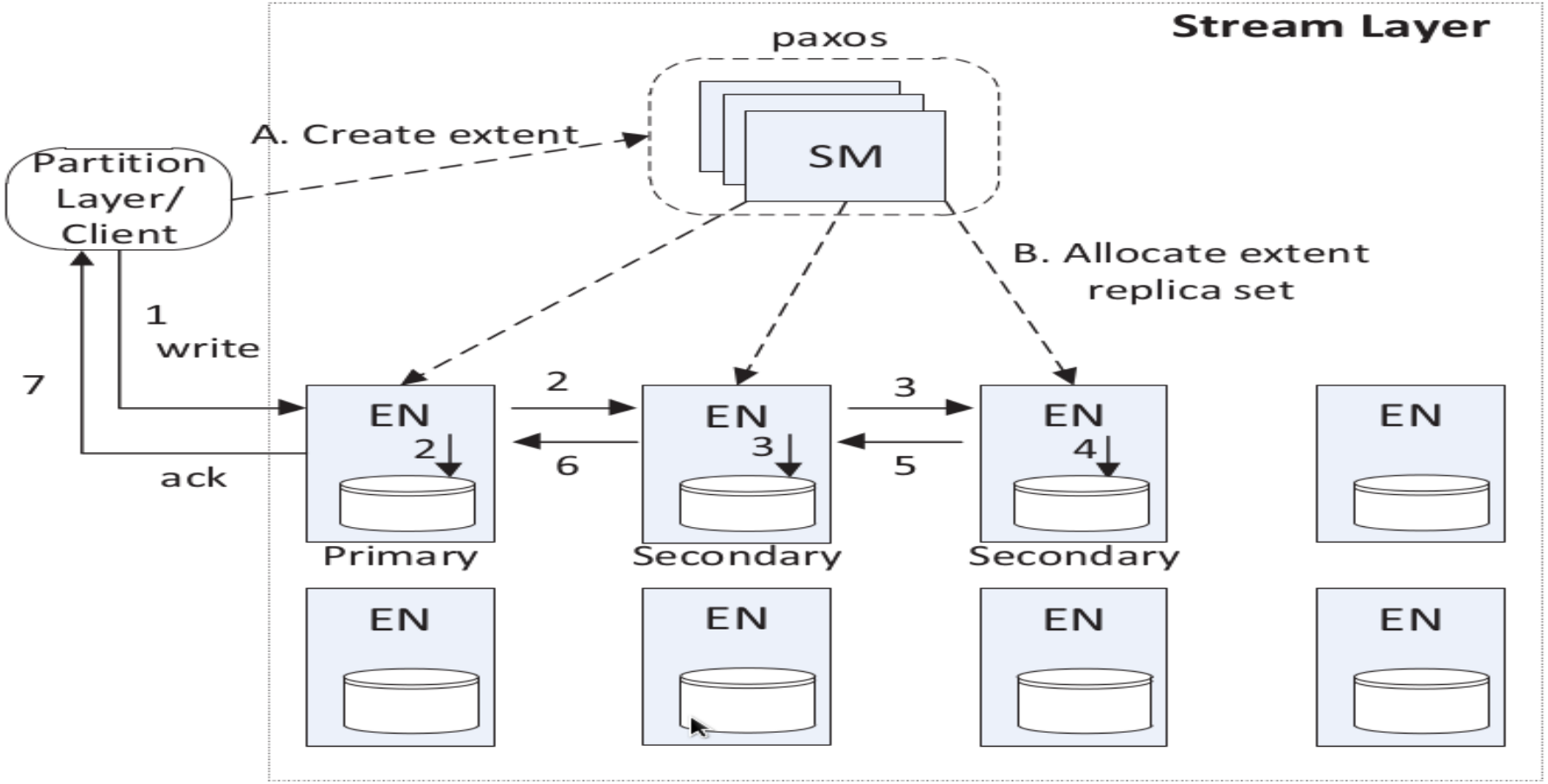
Stream Manager

- Keeps track of the stream namespace, what extent are in each stream, and the extent allocation across the Extent Nodes.
- Performs lazy re-replication of extent
- Monitors health of the Extent Nodes

Extent Node

- Maintains the storage for a set of extent replicas
- Deals only with extents and blocks
- Talks only to other Extent Nodes

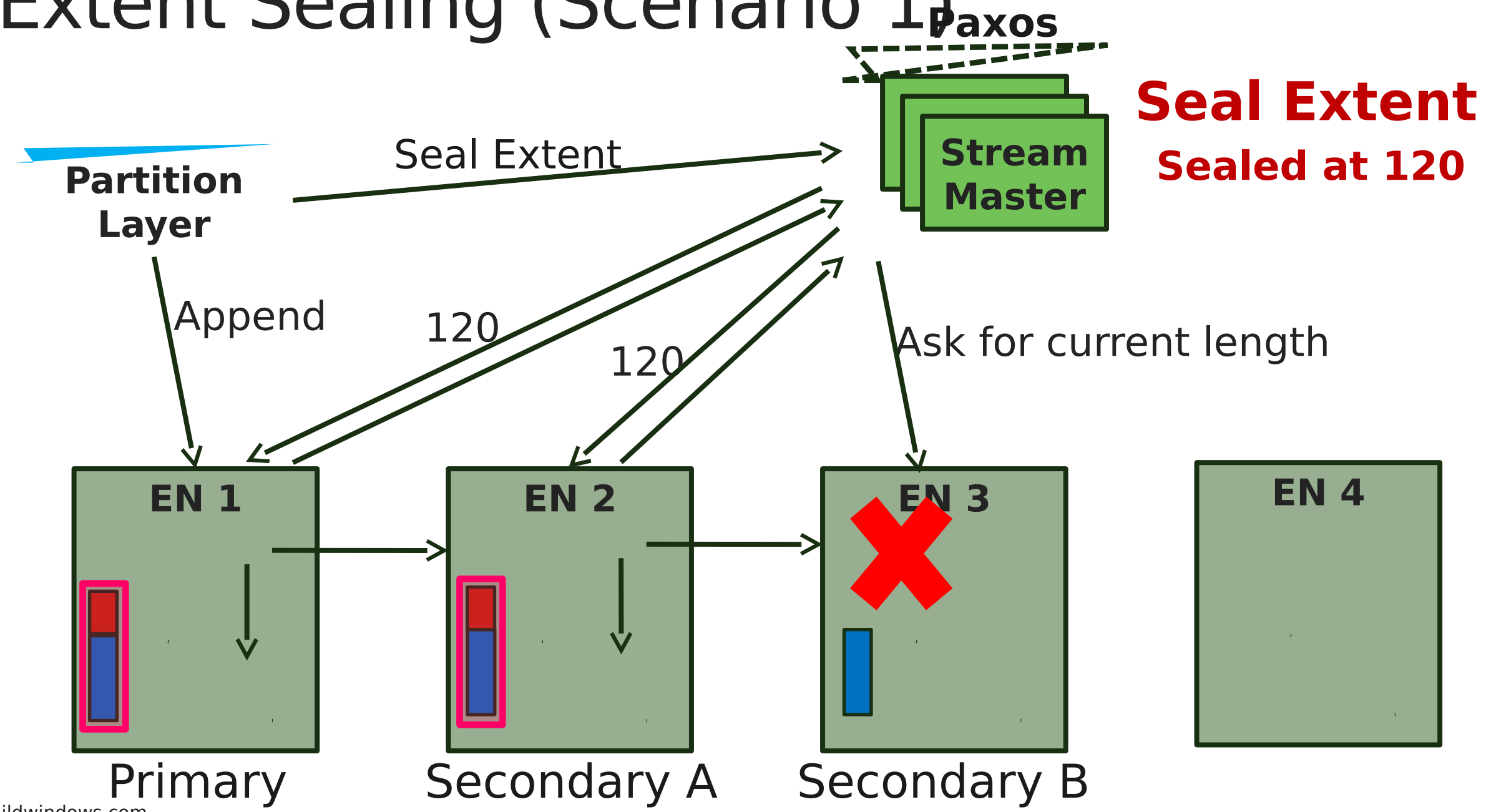
Stream Layer Intra-Stamp Replication



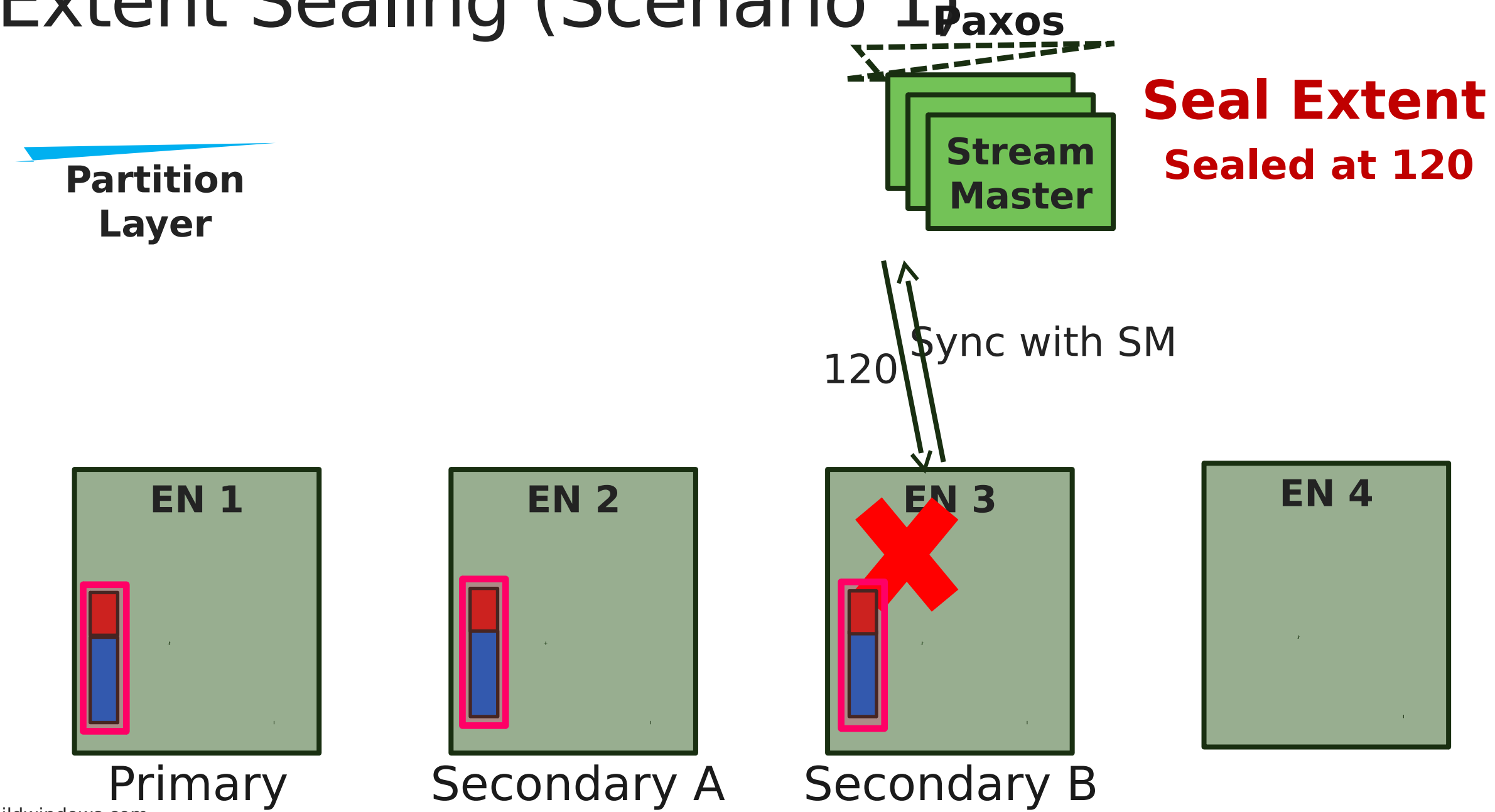
Providing Bit-wise identical replica

- Primary Extent Node for an extent never changes
- Primary Extent Node always picks the offset for appends
- Append for an extent are committed in order
- Sealing strategy

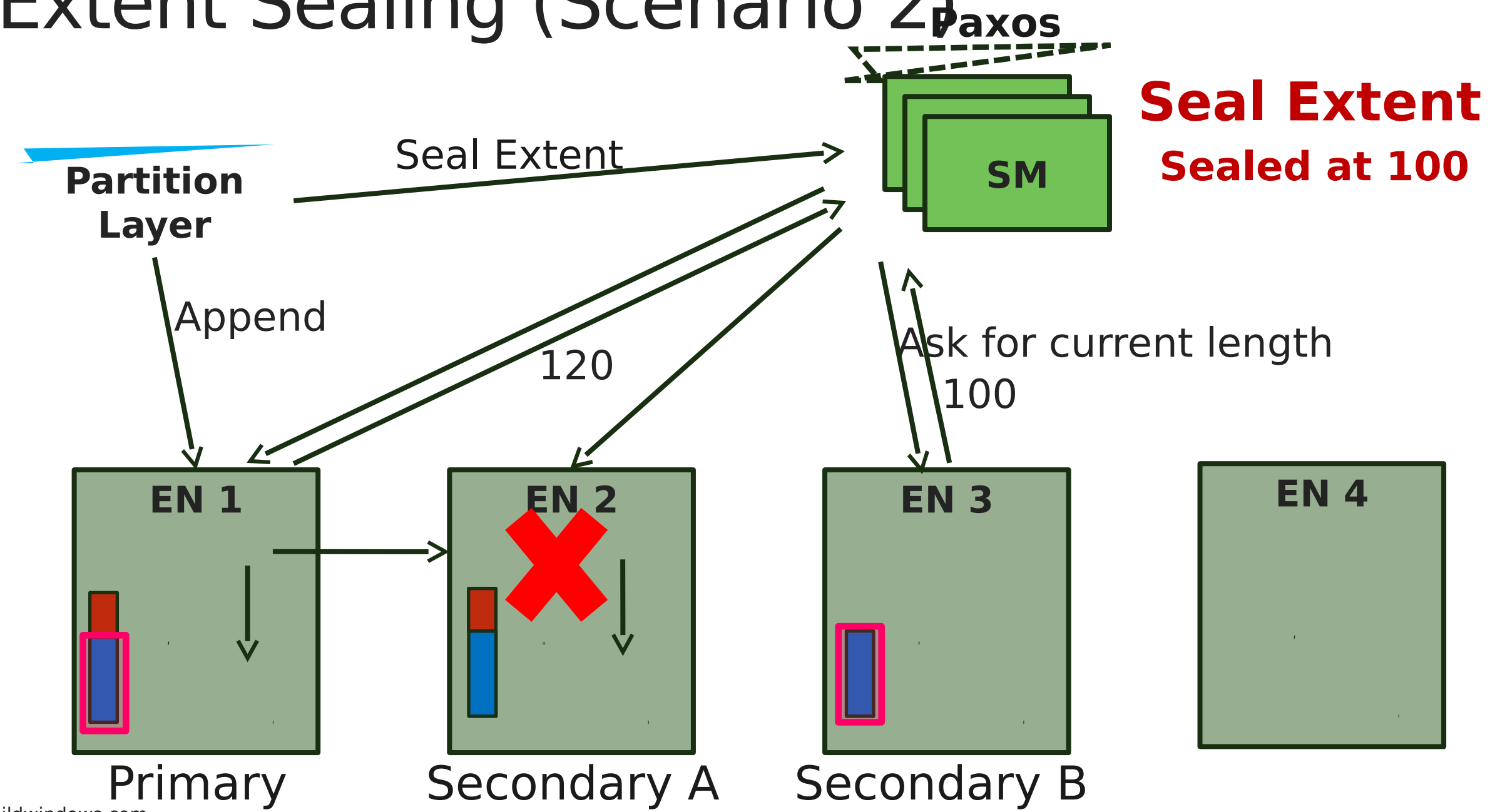
Extent Sealing (Scenario 1)



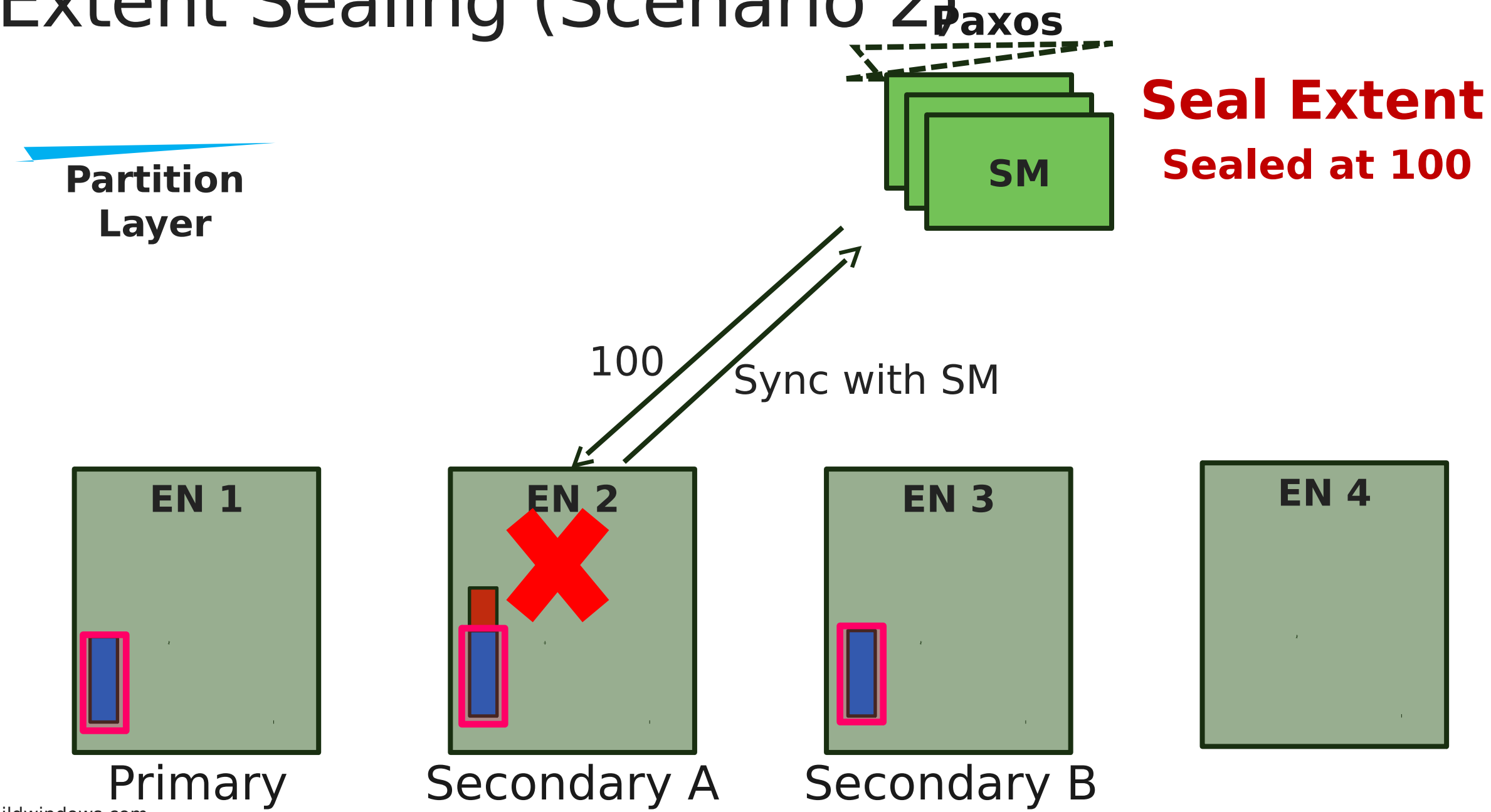
Extent Sealing (Scenario 1)



Extent Sealing (Scenario 2)

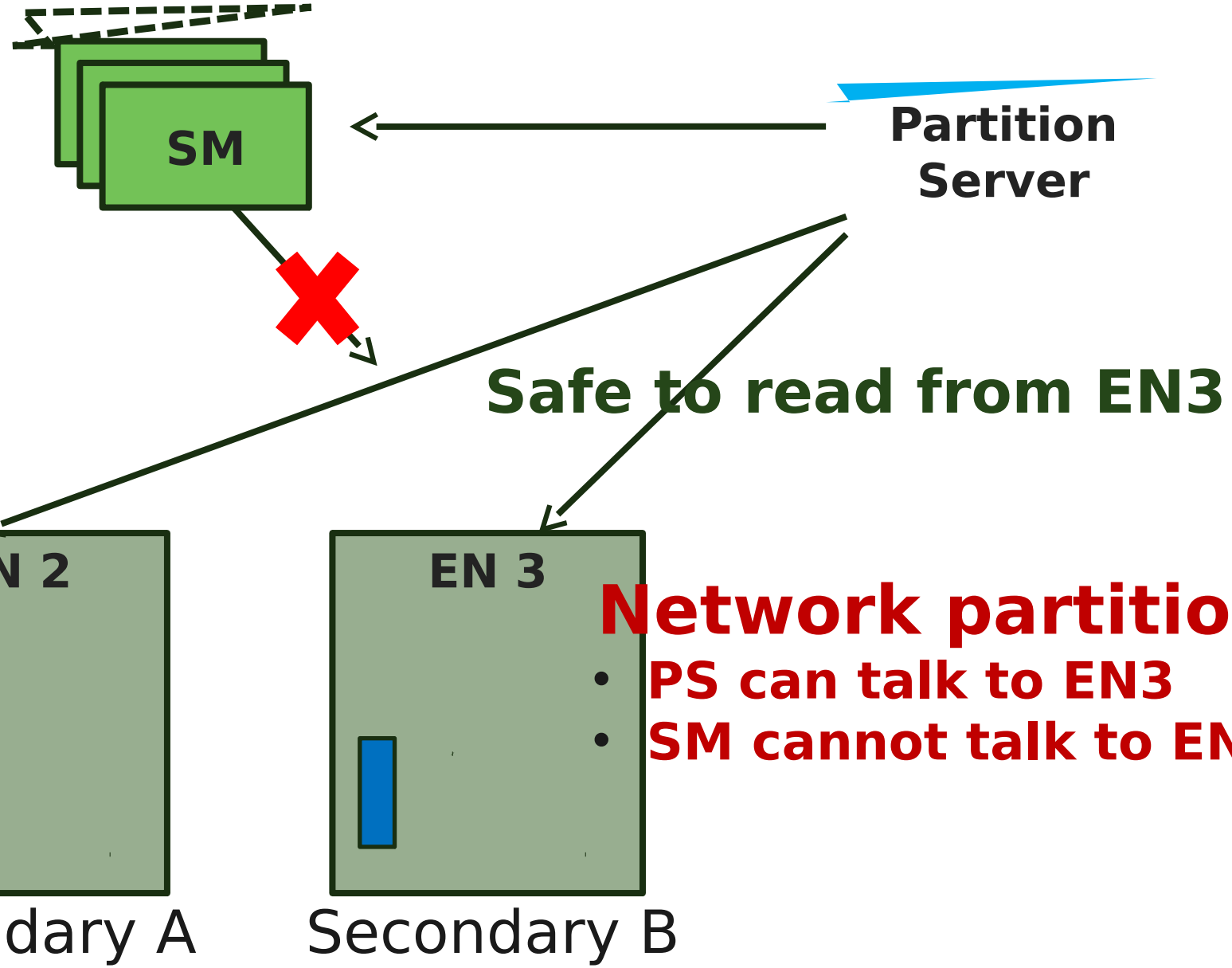


Extent Sealing (Scenario 2)



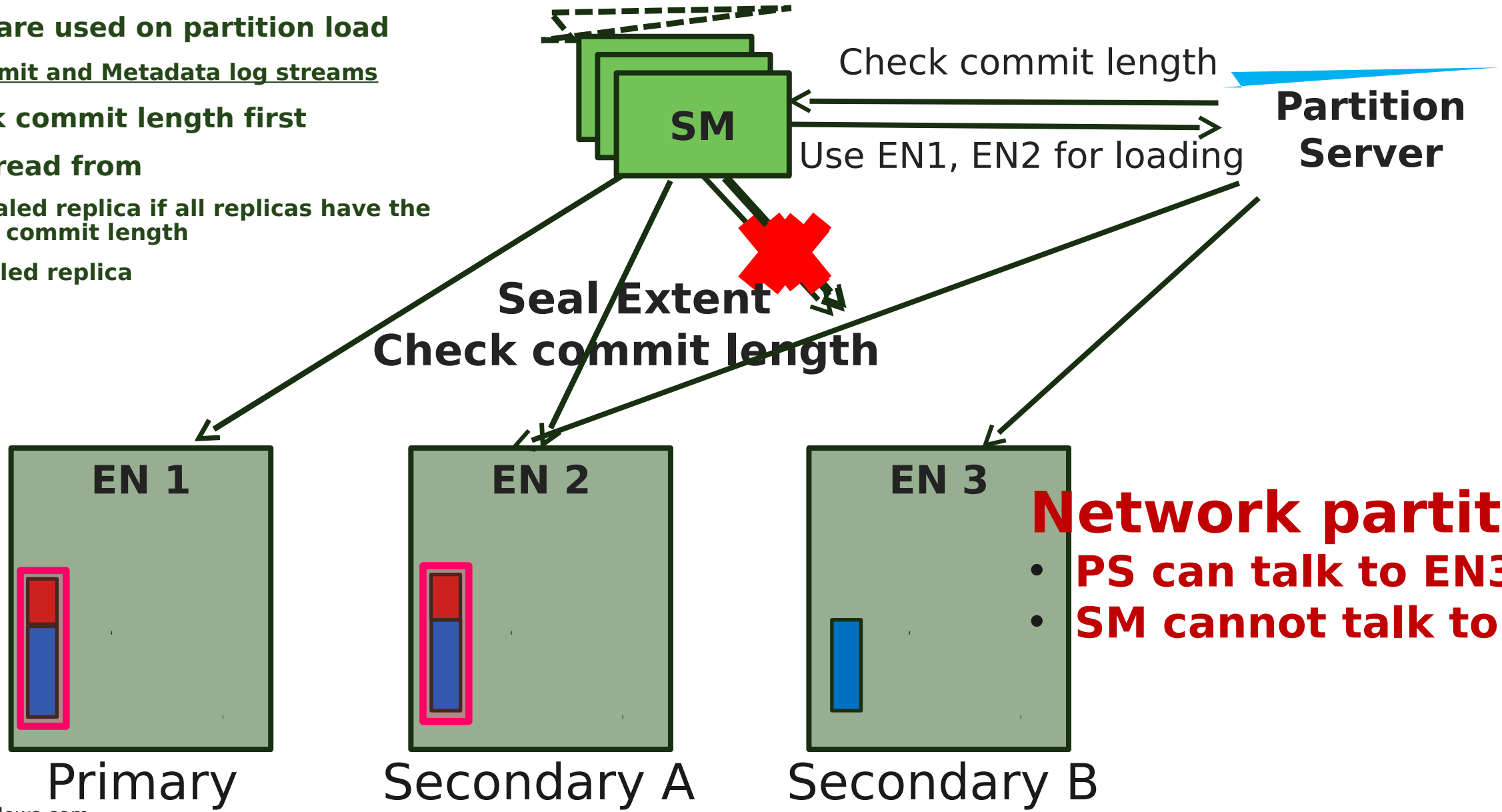
Providing Consistency for **Data Streams**

- **For Data Streams, Partition Layer only reads from offsets returned from successful appends**
  - **Committed on all replicas**
  - **Row and Blob Data Streams**
- **Offset valid on any replica**



Providing Consistency for **Log Streams**

- **Logs are used on partition load**
  - **Commit and Metadata log streams**
- **Check commit length first**
- **Only read from**
- **Unsealed replica if all replicas have the same commit length**
- **A sealed replica**



Network partitio

- **PS can talk to EN3**
- **SM cannot talk to EN**

Durability and Journaling

- Three durable copies of the data stored in system
- On each Extend Node a whole disk is reserved as a **journal drive**
- The journal drive is dedicated solely for writing

Partition Layer

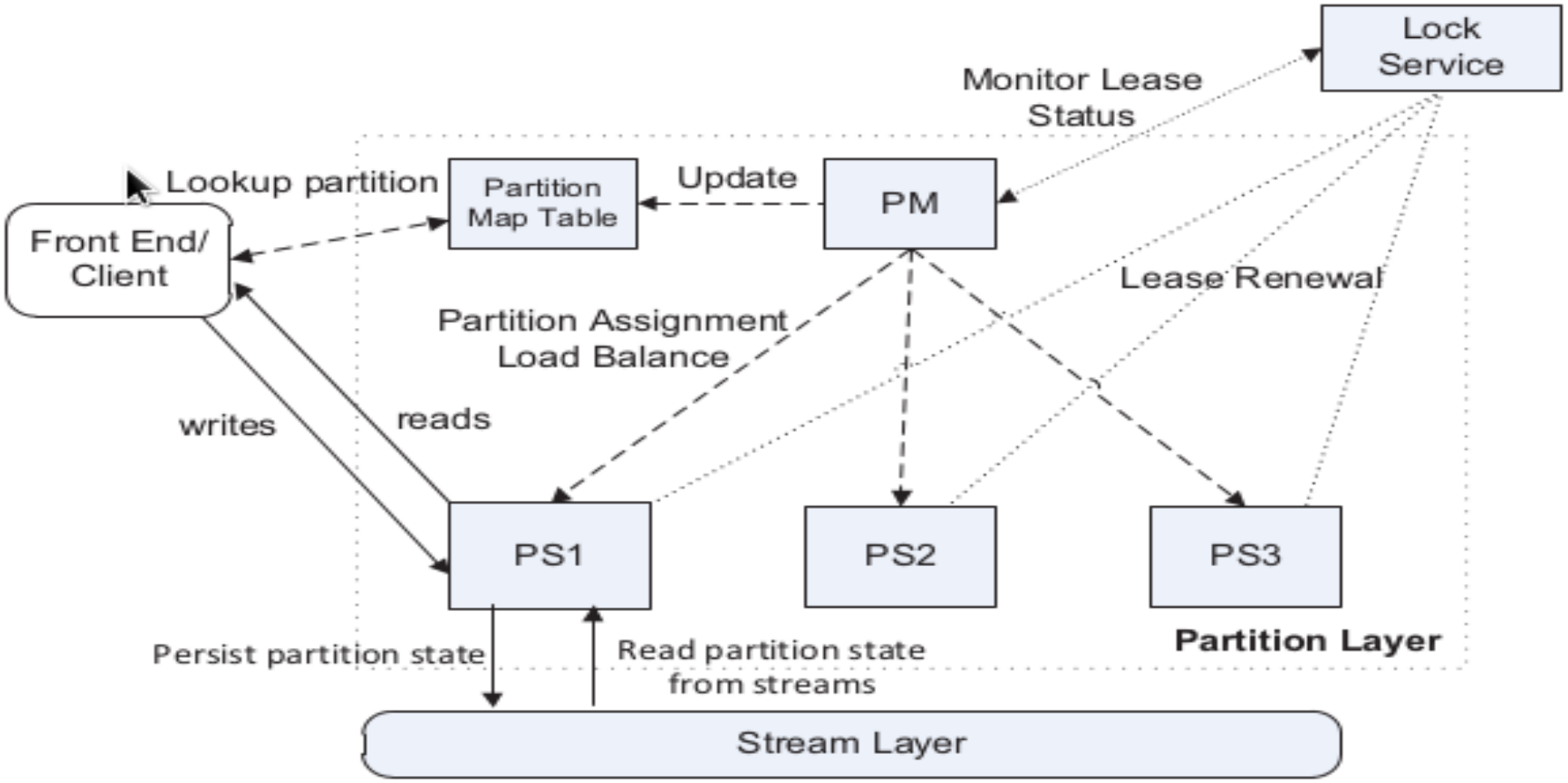
Partition Layer

- Stores different types of objects (blob, table or queue)
- Understands what a transaction means for a given object type
- Spread the index across many servers
- Dynamically load balance

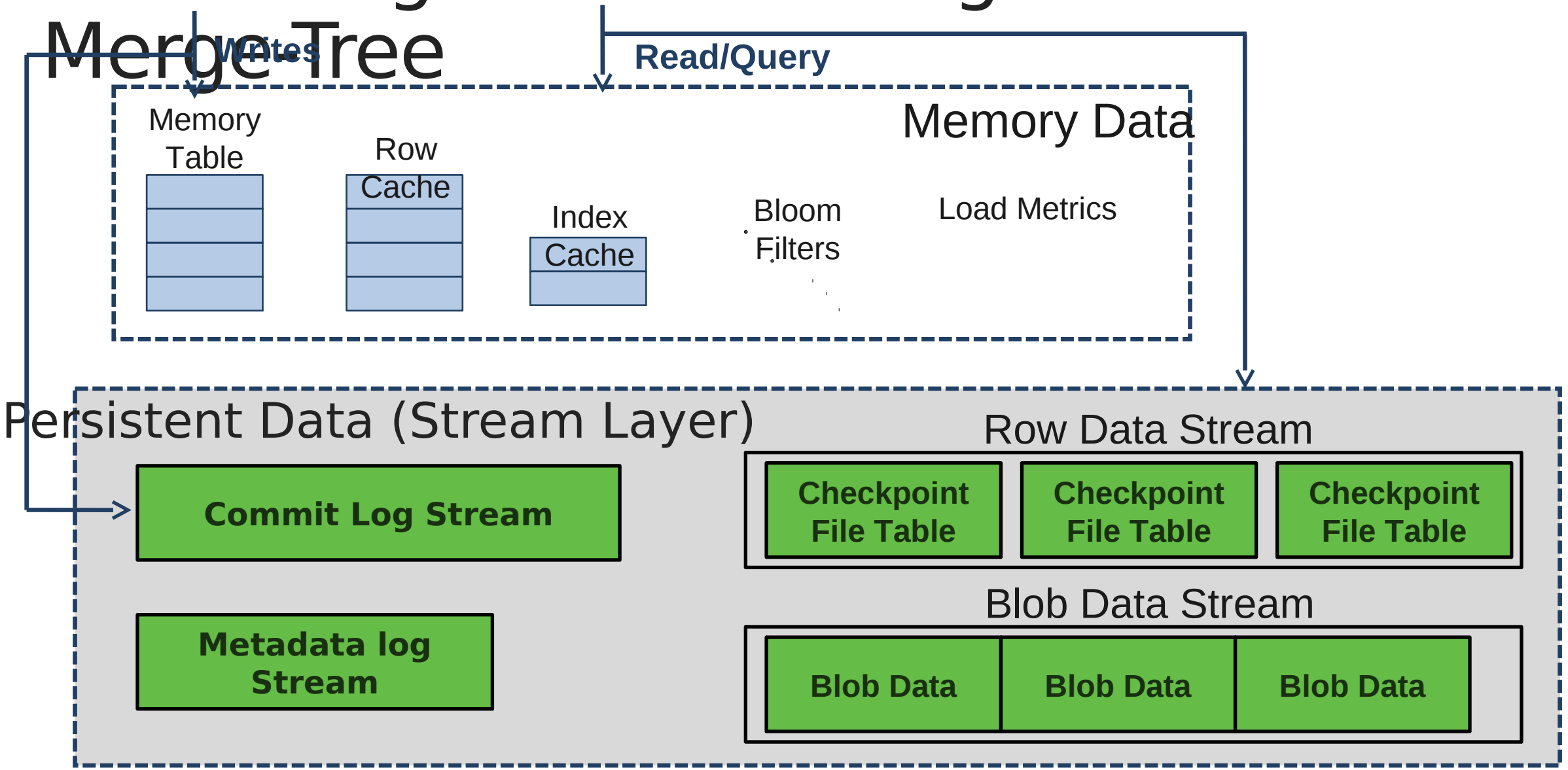
Partition Layer Data Model

- Provides internal data structure called Object Table
  - Account Table: stores metadata and configuration for each storage account assigned to the stamp
  - Blob Table: contains all blob objects for all accounts in a stamp
  - Entity Table: stores entity rows for all accounts in a stamp
  - Message Table: stores all messages for all accounts in a stamp
  - Partition Map Table: keeps track of the current RangePartitions
- Object tables are dynamically broken up into RangePartitions

Partition Layer Architecture



Each RangePartition – Log Structured Merge Tree



RangePartition Load Balancing

- The Partition Manager performs three operations to spread load across partition servers and control the total number of partitions in a stamp:
  - Load Balance
  - Split
  - Merge
- Based on:
  - Transactions/second
  - CPU usage
  - Network usage
  - Request latency
  - Data size of RangePartition

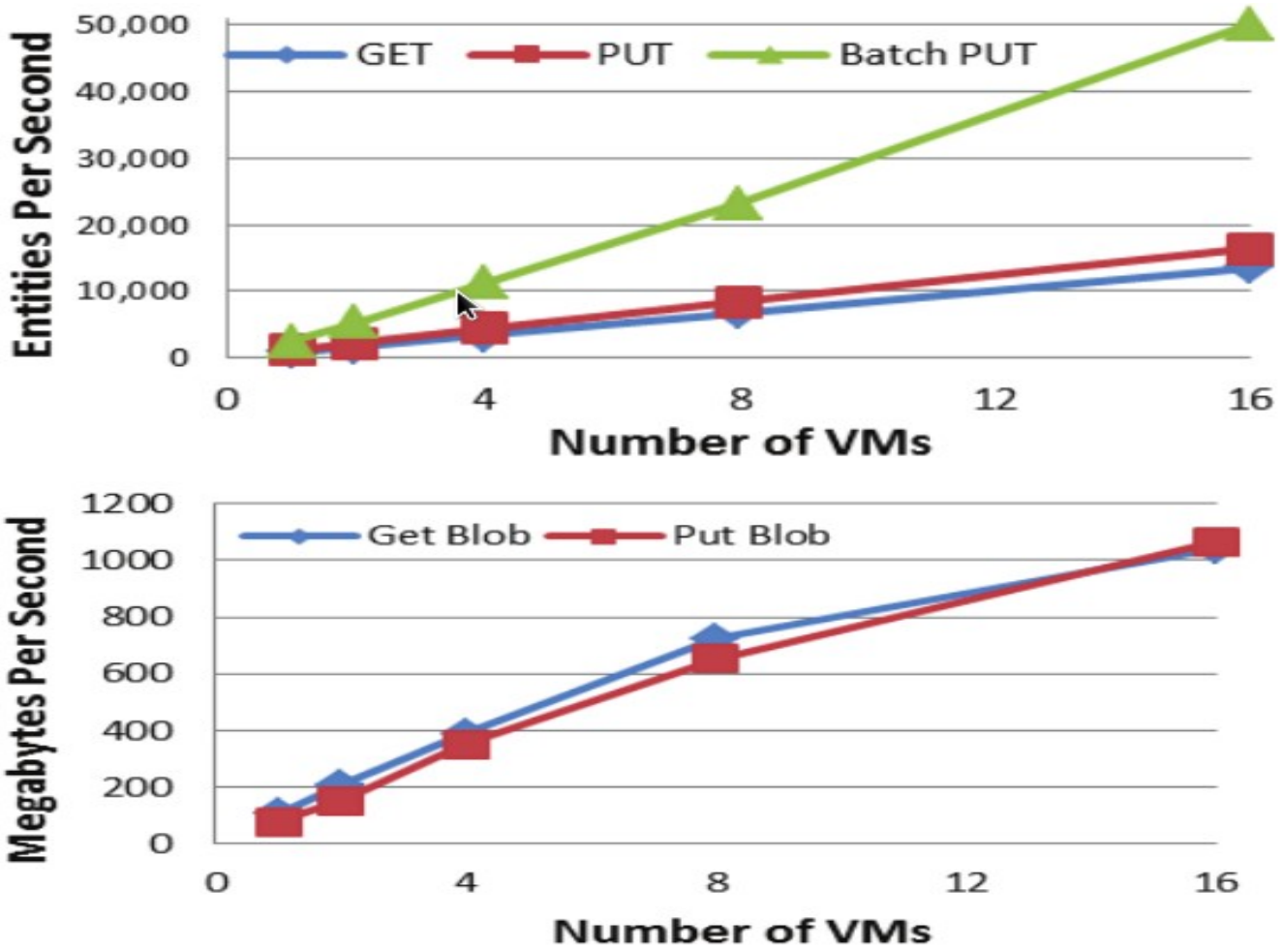
Inter-Stamp Replication

- An account has primary stamp and one or more secondary stamps
- Inter-Stamp replication is done asynchronoulsy
- Disaster recovery and account migration

Application Throughput

- Customers run their applications as a service on VMs.
- Seperate computation and storage into their own stamp
- Examine the performance of a customer application is running from their hosted service VM in the same data center as where their account data is stored

Application Throughput



Workload Profiles

Table 1: Usage Comparison for (Blob/Table/Queue)

		%Requests	%Capacity	%Ingress	%Egress
All	Blob	17.9	70.31	48.28	66.17
	Table	46.88	29.68	49.61	33.07
	Queue	35.22	0.01	2.11	0.76
Bing	Blob	0.46	60.45	16.73	29.11
	Table	98.48	39.55	83.14	70.79
	Queue	1.06	0	0.13	0.1
XBox GameSaves	Blob	99.68	99.99	99.84	99.88
	Table	0.32	0.01	0.16	0.12
	Queue	0	0	0	0
XBox Telemetry	Blob	26.78	19.57	50.25	11.26
	Table	44.98	80.43	49.25	88.29
	Queue	28.24	0	0.5	0.45
Zune	Blob	94.64	99.9	98.22	96.21
	Table	5.36	0.1	1.78	3.79
	Queue	0	0	0	0

Thank you!
Any questions?