MoSMB
SMB with Mojo

Case Study

SMB3 FILE STORE FOR HYPER-SCALE SOFTWARE DEFINED STORAGE

Futuristic SMB3 stack for forward thinking companies

MOSMB (SMB WITH MOJO) IS RYUSSI'S ADVANCED SERVER MESSAGE BLOCK (SMB) SERVER COMPONENT LIBRARY OFFERING FOR THE DATA STORAGE MARKET TO ENABLE FILE SHARING IN A HETEROGENEOUS ENVIRONMENT THAT INCLUDES BOTH WINDOWS AND NON-WINDOWS COMPUTERS.

EXECUTIVE SUMMARY

This case study illustrates the seamless experience in integrating MoSMB to a customer's hyper-scale SDS platform with a proprietary object store interface.

MoSMB design is highly flexible and modular with well-defined interface for different modules. This immensely helped meet the customer's need for an easy integration of MoSMB library with their platform code.

The customer was able to demonstrate SMB3 file server use case in a matter of 60 days! And in just four months, they were able to successfully deploy the Beta version of the integrated SDS product for their Media & Entertainment industry customers.

OVERVIEW

Ryussi's customer has an award winning hyper-scale software-defined storage product platform that runs on x86 servers and dynamically tiered virtual volumes across SSDs and HDDs. The platform's unified connector design provides interfaces for block (iSCSI), file (NFS), object (S3) and Hadoop Distributed File System (HDFS). To tap the Hyper-V market, they were looking to add SMB3 NAS store to their platform.

The primary SMB3 use cases that they were interested in were high performance SMB3 file server and Hyper-V over SMB. The targeted hardware platforms were Cisco UCS, HPE Apollo & Proliant servers.

The customer evaluated various options for incorporating SMB3 and found MoSMB as the best option for their requirements. They also did not have SMB3 expertise in-house and wanted Ryussi engineers to integrate MoSMB with their storage stack.

SOLUTION

The customer wanted to enable the SMB3 use cases by adding the SMB3 connector to their platform's unified connector architecture.

The customer's proprietary data interface abstracted the southbound and inner data model of their virtual storage platform from the outside application access layer.

The SMB3 protocol connector was developed using this data interface. In order to integrate MoSMB into the customer storage stack two primary integrations had to be done at the code level.

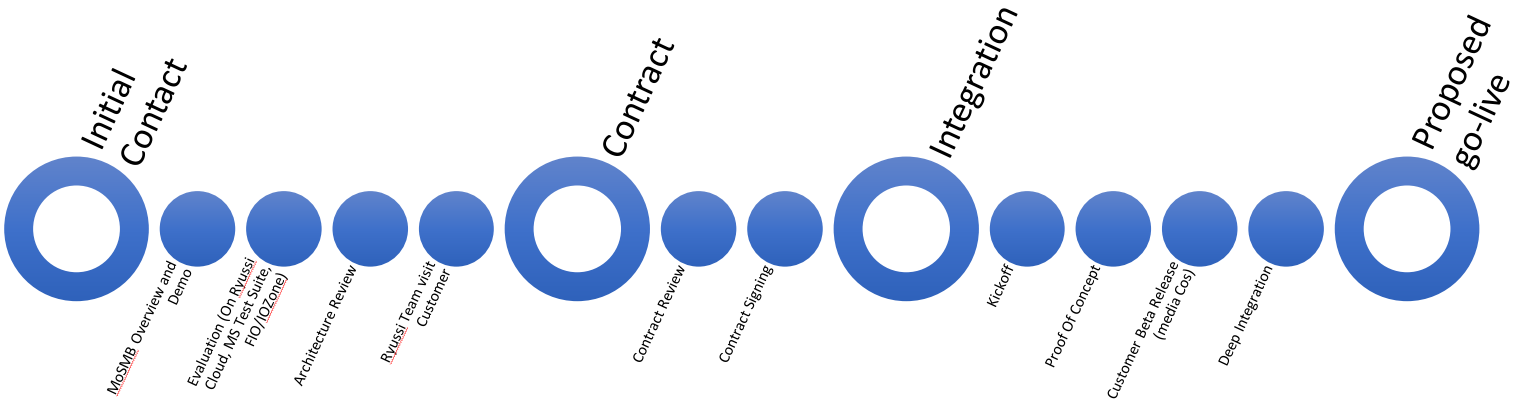
1. A connector had to be built which would load the MoSMB libraries and initialize them appropriately. This connector would interact with customer's configuration manager to get the required configuration information.
2. An IOS driver had to be built which would use the C language bindings for the proprietary data interface to do the 'file' level metadata and data operations. These operations would comply with the NFS idea of a 'file'.

Additional integration points were to be developed for logging systems, statistics and performance counters, Kerberos based authentication, debugging and tracing and dynamic addition and deletion of shares.

ENGAGEMENT MODEL

A Ryussi architect visited the customer's US office to understand their storage platform and create the integration plan. Ryussi's remote engineering team worked remotely to integrate the solution as per the customer requirements.

The initial phase involved standalone SMB3 server addressing the file server use case for their customers from the Media and Entertainment industry. The high availability aspects using MoSMB's Witness server were to be addressed in a later phase following the availability of the customer's clustering interface.



KEY INTEGRATION AND DEVELOPMENT TASKS

The MoSMB connector was developed with MoSMB start and shutdown methods as the entry and cleanup points. The start method called the configuration manager functions to obtain the configuration data and set up the MoSMB ecosystem appropriately including the Kerberos configuration. The configuration information also provided the IP addresses/Virtual IPs on which the transport interface would start receiving the SMB packets.

The customer's storage model was comprised of objects and containers (a set of objects). The container was mapped to the traditional file system entity. It contained a metadata object and data objects. The traditional namespace hierarchy was built atop the flat container space by having the mapping stored in a distributed replicated NoSQL database.

This model of a file and mapping it into the semantics expected of an underlying store by the SMB protocol is accomplished in MoSMB by the IO Subsystem (IOS) driver. MoSMB supports the integration of custom storage integration IOS drivers in the form of dynamically loaded libraries.

The following integration tasks were undertaken:

1. Develop the MoSMB IOS driver for virtual volumes over the platform's data interface using C language bindings.
2. Have the file to object mapping compatible with the existing file to volume mappings used by NFS.
3. Integrate MoSMB into the customer storage stack as a file based connector akin to the existing connectors.
4. Have the system work in conjunction with an Active Directory based Microsoft Windows domain. Verify SMB functionality using Hyper-V and Windows Explorer as clients of the system.

To know how you can use MoSMB as a File Server on your platform, please contact sales@ryussi.com