OS Considerations for SRP on IB

The combination of IB and SRP represents a number of novel implementation problems. Unlike PCI and other contemporary bus technologies, IB is an interconnect technology simultaneously accessible by multiple systems. SRP provides a standardized SCSI encapsulation and communication protocol that's independent of the actual storage interconnect technology (for example, SRP should be that same for all IB implementations whether the storage is attached through FC, SCSI or IP). SRP on IB provides an unprecedented level of commonality.

In this situation, it's tempting to try to follow a much more restricted technological precedent (such as using IDE as a model for SRP) or to fail to generalize functionality from other technologies (like making the SRP protocol specific to each storage bus). Either one of these extremes would require continual updates to the standards to support different storage buses, new features and alternative approaches resulting in poorly performing, expensive implementations.

Another potential pitfall is to fail to underestimate the robust support provided by an OS in this type of environment. A consequence of this can be the specification of redundant functionality that not only duplicates OS features but actually hinders how they work. This is particularly regrettable because duplicating OS functionality can be very expensive for hardware vendors to implement, test and maintain while creating few opportunities for product differentiation.

For SRP to be successful, it helps to understand the environment where it will be used and the features needed there.

Basic IB Features

For OS support, all configuration operations on IB should be regarded as event driven. A node is identified, determined to support SRP, an SRP driver is notified and establishes communication. There doesn't have to be a table on every system, nor does there have to be a free-for-all among all the systems. In fact, the mechanism that determines how an SRP node is associated with a particular system is OS specific. It's important that an SRP implementation be entirely independent of any particular association mechanism because this is an area that is likely to evolve quite significantly over time.

An SRP implementation only needs to utilize a small number of management datagrams (TBD) to identify it as an SRP controller. Clearly these are essential because the idea that it will somehow be implicitly known that a particular IB node supports SRP doesn't support an event driven mechanism.

Implementation Differentiation

Even though SRP is independent of the technology used to connect to the drives, it doesn't seem particularly realistic or helpful to hide this information deliberately from the OS. It is probably quite useful for the OS to be able to distinguish whether the SRP implementation is fronting a RAID box, an FC bridge or an iSCSI bridge. There are probably sets of attributes that are specific to each type of storage bus technology and there are certainly vendor specific attributes that need to be accessible to the OS. Another possible set of attributes could reflect the SCSI command set available through SRP (tape drives and their drivers are significantly different than disks).

It seems that there could be four levels of attributes associated with an SRP implementation: generic SRP, storage bus, command set and vendor specific. IB doesn't really have an effective mechanism for expressing this sort of thing. Ideally, attributes would be somehow accessible through the IOC Profile but that's not really flexible enough for this. An SRP specific attribute retrieval MAD might not be inappropriate.

Given the different “levels” of attributes and a tendency for them to proliferate over time, there probably isn’t a reasonable way to predefine them all and structure them. 1394 and SBP-2 use a P1212 based Configuration ROM to support these types of attributes and a similar format could be used to define the contents of an SRP attribute MAD.

IOC Differentiation

As mentioned previously, IB configuration within an OS is event driven and the process of configuration is highly OS specific. Even so, an SRP implementation needs somehow to uniquely identify the resources it provides to the IB fabric. Since IB is a multi-hosted technology, most SRP implementations will likely support “connections” to multiple systems. In addition, a particular “system instance” could be moved to a different IB address in the fabric (for example, replacing a faulty system node) or the storage resources of a particular “system instance” might need to be accessible by a different “system instance”.

A very simple approach to this problem is for an SRP implementation to support multiple IB IOC profiles where each one provides access to a subset of the LUN’s (or should that be targets?) available to that implementation. How these subsets are defined is currently outside the scope of this document. It’s almost definitely dependent on the storage bus technology; it could very well be vendor specific; and the implementation could even do things like dynamically remap LUN values as they pass through it. The essential point is that when a connection is established to an IB SRP IOC, there is a filtering function that limits access to the storage associated with that IOC.

This vastly simplifies the IB configuration problem. Because there is a GUID in each IOC profile, associating a particular system and a storage pool is reduced to matching the GUID. Since configuration is driven by the OS, if the system address changes, there is no impact on the IOC.

This mechanism isn’t intended to preclude or block configuration mechanisms that might be associated with the storage bus technology but it provides a configuration mechanism that’s storage technology agnostic.

Summary

SRP on IB presents several challenging problems but it’s relatively easy for an OS to handle them if the SRP implementations provide a few simple configuration mechanisms. Providing easy identification of storage related attributes and using the IOC GUID to determine system associations supports an event driven IB configuration model. This involves minimal complication to an SRP implementation while allowing an OS to adapt new configuration practices.