Why In-Place Copy Data Management is The Better Choice

Copy Data Management (CDM) is increasingly being recognized as a must-have technology for the modern data center. But there remains a fundamental difference in approach among vendors: should you use In-Place CDM or go with an Out-of-Band, appliance-based model? Let's look at the differences, but first let's agree on what CDM means in the first place.

What is Copy Data Management?

Copy Data Management is managing the creation, use, distribution, retention and clean-up of data copies. Different feature sets may or may not be added on top of the solution; things like automation, user self-service options, REST APIs for DevOps integration, reporting and so on (Catalogic offers all of these features).

The goal of CDM is to eliminate the many manual steps currently used to create and deliver copies – like getting rid of scripting – and to centralize operations into one tool rather than several. The goal being greater efficiency in storage consumption and – much more importantly – dramatically lowering operational overhead and improving reliability.

There are two main architectural approaches to CDM: the Out-of-Band method and the In-Place method. Let's look at them.

The Out-of-Band Approach

In the Out-of-Band (OoB) model, data is moved from the source production storage to a target storage device; that is, a separate storage stack different from the primary. The data is typically moved by a client agent residing on the host server. The copy method is usually some kind of changed-block model. While this reduces the host impact compared to traditional backup, the impact is not zero as it will affect CPU utilization and host bandwidth.

The target device is usually an appliance, though some vendors offer "bring your own storage" software versions as well in the form of a virtualized appliance. It is important to realize that the appliance is a separate storage device with its own controller and firmware. This can be significant and will be discussed further below.

When using the data copies (for reporting, dev-test, etc.), a workload server connects to the OoB appliance. If changes need to go back to the production side, they would be copied back to production over the network. Multiple workload servers can connect to a single copy, which is parceled out via snapshot images. This is how efficiency benefits are realized. But performance characteristics of the OoB appliance of course will be different than the production storage.

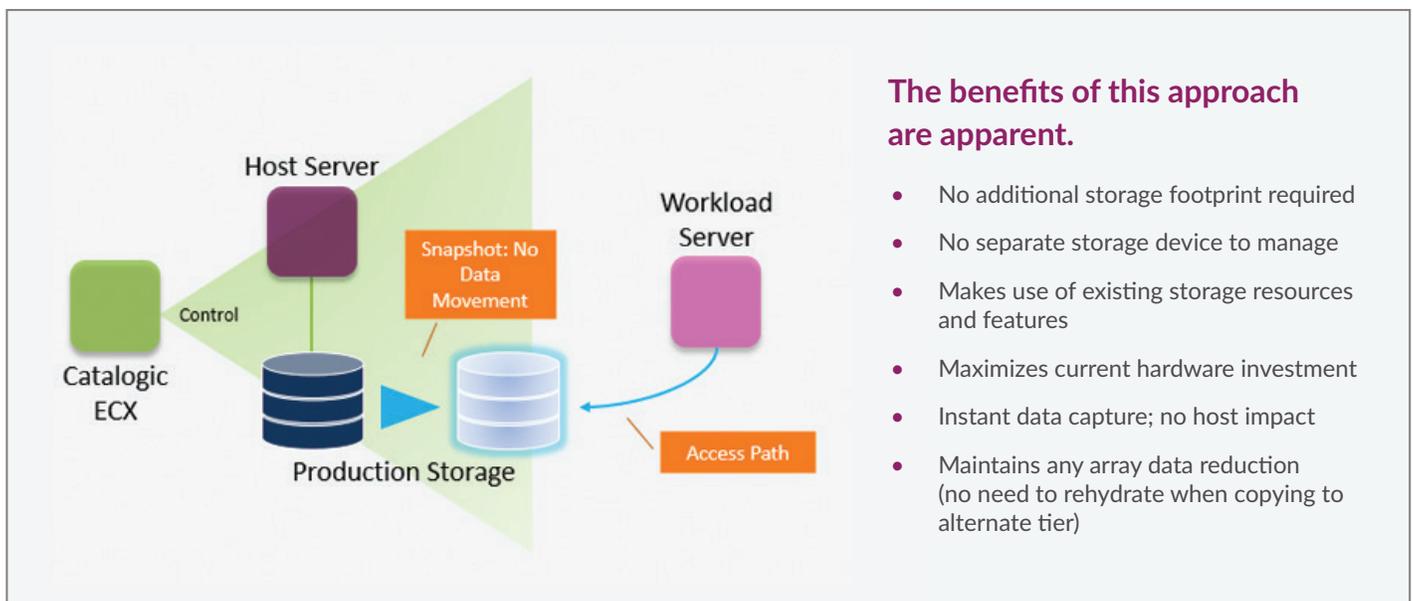**The Out-of-Band approach can be visualized as follows:**
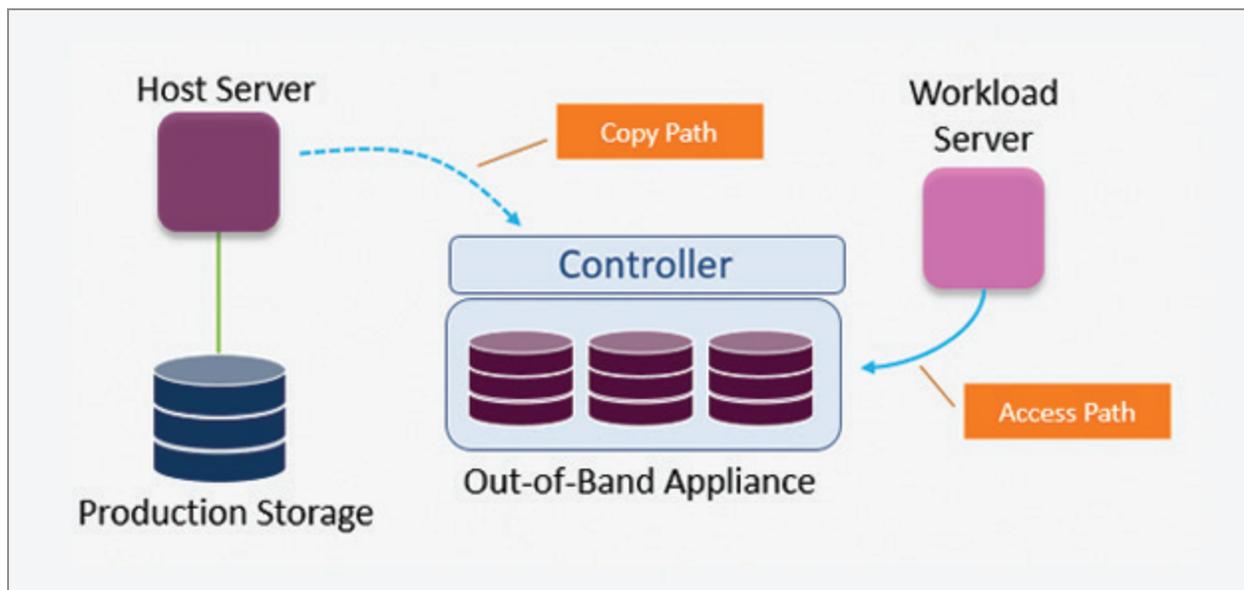
The In-Place Approach

Catalogic uses an In-Place Copy Data Management model. This In-Place approach makes use of primary storage array copy processes – snapshots, replication and clones – as well as any data reduction inherent in the array (deduplication, compression). Catalogic ECX functions as a *control plane* and doesn't replace any storage array features; it makes use of them, providing features like automation, self-service, application integration (for data consistency, log management, etc.), hypervisor integration, and so on. There is no separate storage device involved. Catalogic communicates to both the host server and the production storage, driving copy processes.

In this model, there is no data movement as storage snapshots are created In-Place. Workload server access is provided by using the storage cloning features and connecting the snapshot clone to the server. With the In-Place approach, recoveries or data copying are far quicker because little or no data movement is involved, and nothing traverses the network.

This approach can be visualized as follows:

The benefits of this approach are apparent.

- No additional storage footprint required
- No separate storage device to manage
- Makes use of existing storage resources and features
- Maximizes current hardware investment
- Instant data capture; no host impact
- Maintains any array data reduction (no need to rehydrate when copying to alternate tier)

There is one aspect of the In-Place model that should be discussed. Not everyone is comfortable with running non-production workloads on production storage. Even though all-flash arrays have plenty of IOPs to spare and can run non-prod workloads without performance impact (unlike with spinning disk), many organizations simply won't allow non-prod work to happen on prod storage.

Catalogic provides an easy solution for that. Since Catalogic ECX can manage replication as well as snapshots, it's simple enough to replicate to an alternate array (which can be in the same storage rack or physically separated) and run non-prod workloads on the alternate array. **This is visualized as follows:**
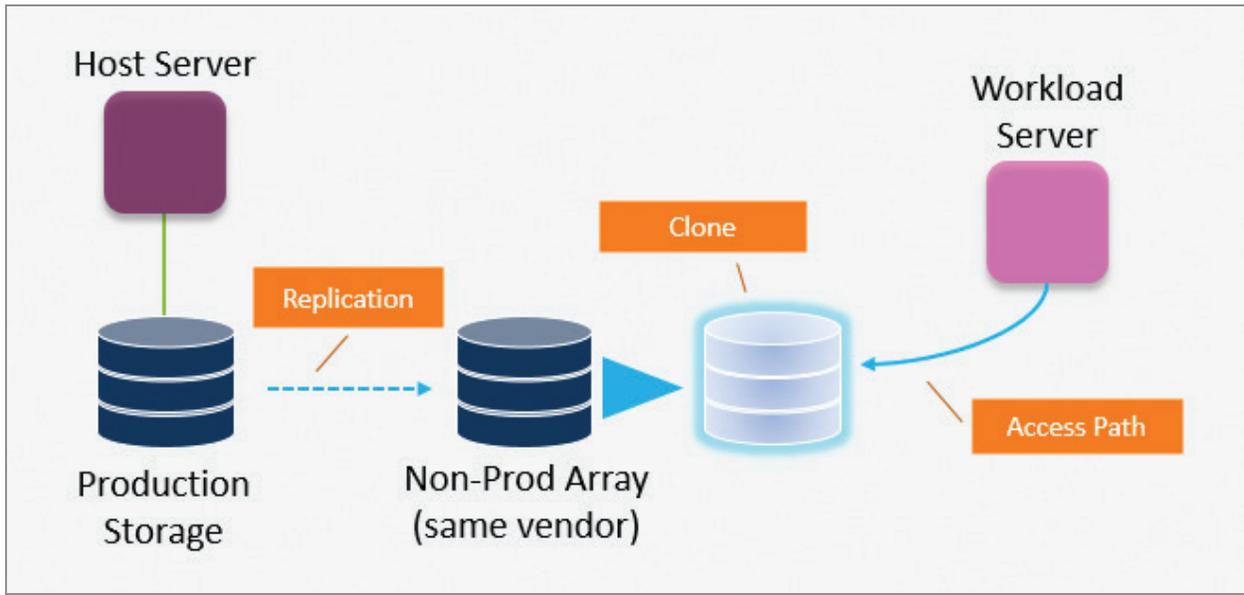
In this way, you have complete array separation. A major benefit of this approach is that by using array replication, you maintain all the data reduction from the primary array, thereby greatly decreasing the amount of data replicated. And no data goes through the host, which limits network and host impact and reduces complexity.

There's something else to consider, as we learned from end-users. Consider the case of Dev-Test. In the Out-of-Band model, your Dev work is being done on a storage system that is *fundamentally different than your production storage*. It has different software, a different file system, different disks, which all equate to different performance characteristics. Therefore, if you are doing development work on a different system, when you move the application back to production *you can't be certain it will work properly*. An element of risk remains. And when something does go wrong, it inevitably leads to finger-pointing as development blames IT for an infrastructure problem and IT blames development for problems with their code. Troubleshooting this kind of problem can be nightmarish, because the source of the problem could be anywhere on either side of the equation.

In the Catalogic approach, the same array type is used in prod and non-prod, so all your Dev work is done on a storage stack that is *100% identical to production*. Even the copy processes use the native array software. Catalogic *does not introduce any software changes*; it only manages the process by talking to the array APIs. This is a no-risk infrastructure model for development.

Conclusion

In some ways, this choice in approaches is reminiscent of the early days of VTLs and deduplication. There was a huge battle about the better approach: inline dedupe during backup, or post-processing dedupe after backup completes. The battle raged for a few years, until hardware improvements made it impossible to argue anymore: in-line dedupe won, because it was clearly more efficient. A similar shakeup is happening in the copy data space. Out-of-band was the first approach that came to market, but In-Place is proving to be the better choice.

The differences are straightforward. The Out-of-Band model requires you to deploy an entirely different storage hardware/software stack with different operational and performance behaviors. And incidentally, the appliances are often quite expensive. But the costs are more than the initial purchase price. You must manage and maintain a separate storage environment just for the sake of your copies. And you also have to consider scale. Catalogic has seen users with thousands of database instances, each of which spins off 8 or 10 copies for non-prod workloads.

How many copy appliances will you need to deliver on this kind of workload demand?

Catalogic's In-Place approach lets you maximize the storage investment you've already made (investments in both money and operational know-how). If required by budget limits and allowed by policy, you can run non-production workloads on the production array. That is, with an easy, non-disruptive software tool you can bring automation, self-service and other operational enhancements to your existing storage footprint. If you need to deploy a second array (or third, fourth, fifth...), Catalogic fully manages operations across nodes: production array snapshots for recovery (scheduling, retention, easy recovery workflows, etc.) and replication (scheduling, retention, easy use of data at the second node, etc.) to put data on a separate frame.

Your choices in a copy data management approach are clear, and we think the better choice is clear as well.