Policy-Driven Management of Data Sets

Jim Holl, Kostadis Roussos, and Jim Voll – Network Appliance, Inc.

ABSTRACT

Contemporary storage systems separate the management of data from the management of the underlying physical storage media used to store that data. This separation is artificial and increases the overall burden of managing such systems. We propose a new management layer that unifies data and storage management without any loss of control over either the data or the storage.

The new management layer consists of three basic entities: data sets, which describe the data managed by the system, policies that specify rules for the management of data sets, and resource pools that represent storage that can be used to implement the policies for the data sets. Using data sets and policies, data administrators are able to directly manage their data, even though that requires them to indirectly manage the underlying storage infrastructure. The storage administrator retains control over what the data administrator can do to the storage.

This new management layer provides the infrastructure necessary to build mechanisms that automate or simplify many administrative tasks, including the necessary coordination between the data and storage administrators. We describe Protection Manager, the first tool that uses this management layer to present storage management in the context of data management tasks that the data administrator wants to perform, and evaluate the effectiveness of using this management layer as a way to automate backups.

Introduction

Data management, in the form of the management of files, file systems and structured data, has traditionally been a separate discipline from the management of underlying storage infrastructure. Data administrators have historically concerned themselves with the redundancy, performance, persistence and availability of their data. The storage administrator has focused on delivering physical infrastructure that satisfies the data's requirements. Typically, the storage is configured and then, within the constraints of the configured storage, data management takes place. If the storage requirements of the data change, the data must be migrated to different storage or the underlying storage must be reconfigured. Either process is disruptive and requires multiple domain-specific administrators work closely together.

The separation of administrators is a natural outcome of the incompatible and inflexible infrastructures deployed. Consider a small aspect of data management: redundancy. Owners of data need a certain amount of redundancy for two reasons. The first is to tolerate faults in the physical media, such as disk failures or storage array failures. The second is to tolerate human and software errors that occur during the manipulation of the data. The traditional solution has been to use a variety of incompatible storage platforms from storage vendors to provide differing levels of hardware reliability and to use tape for recovering from human and software errors. In effect, the underlying infrastructure acts as a ''proxy'' for managing the data.

Modern virtualization features, such as space-efficient replication, result in a storage infrastructure that eliminates much of the incompatibility and inflexibility found in traditional storage environments, but data management and storage management remain separate disciplines. For example, an administrator for a storage system manages FlexVol [3] volumes and aggregates for provisioning storage and uses SnapMirror [10] and SnapVault [4] for making copies of the FlexVol volumes. A data administrator, on the other hand, manages files or structured data and thinks in terms of copies of their files or structured data. Mapping data management requirements onto storage management primitives still requires human interactions and complex processes. As a result, even with a flexible and homogenous storage infrastructure, data management is still done as if the infrastructure was inflexible and incompatible.

To address the gap between what a storage administrator manages and what a data administrator wants to manage, we introduce a new data management framework that we believe can become the basis for a new unified storage and data management discipline. The data management framework consists of a set of three new management objects: data sets, resource pools, and policies. It also consists of several infrastructure components: role based access control [5], a conformance engine, and a centralized database that holds the definitions of these objects.

The data set represents a collection of data and all of the replicas of the data. A data set is independent of any particular physical infrastructure that is currently being used to store its data and is the basic entity that a data administrator manages. A resource pool is a collection of physical storage resources managed by storage administrators. A policy describes the desired behavior of the data. Role-based access control (RBAC) is used to control access to all managed entities including data sets, policies and resource pools. The conformance engine is responsible for monitoring and configuring the storage infrastructure such that a data set is always in conformance with its policy.

Using this machinery, a storage administrator controls how storage is used by defining specific policies and controls the rights to use these policies with RBAC. Once a storage administrator has configured the policies and access control, data administrators can create or administer his or her data sets by assigning policies from the appropriate authorized set. Because changes in policies allow reconfiguration of the storage, and this reconfiguration is done automatically using the conformance engine, a substantial increase in efficiency results when compared with systems in which management and data management are separated.

For example, in the case of redundancy, the storage administrator first configures resource pools with varying kinds of physical redundancy, such as RAID levels or failover storage capabilities. The storage administrator then constructs policies that can be used to create datasets with varying degrees of physical and backup redundancy. When a data administrator requires storage for a data set, he or she selects a policy that provides the appropriate levels of redundancy. The conformance engine will then provision the actual storage required on the appropriate systems and replicate the data. The conformance engine also monitors the storage to handle scenarios in which the data set is no longer in conformance with the selected policy.

The first implementation of this framework, called Protection Manager, addresses the particular problem of data replica management. The details of this implementation are provided in later sections of this paper.

In the rest of this paper we describe the architecture of the data management framework and how the concepts of data sets, policies and resource pools combine to solve data protection and provisioning problems. We then describe how the machinery was implemented in the Protection Manager tool. We evaluate the extent to which the concepts and framework empower data administrators and simplify administrative tasks as policies change. Finally, we describe future work and make comparisons to other systems.

Related Work

We found a variety of research and systems that targeted aspects of our data management framework, however, none propose or describe an architecture that unifies storage and data management. In particular, none described how storage administrators could securely delegate storage management to data administrators such that global policies are enforced and maintained. Although some research alludes to the need for a common language between the storage and data administrator we are the first to propose a useable model.

The notion of separating a logical view of data from a physical view in storage is not a new concept. Work done at IBM by Gelb describes a similar model where one of the primary goals was to isolate physical device characteristics from application awareness in homogenous IBM environments [6]. The IBM work recognized the need for automation as a key to managing more data effectively, but appears to concentrate on initial provisioning operations, rather than automating any reconfigurations of storage due to policy changes.

Keaton, et al. propose a model for automating data dependability that is very similar to our data management framework [8]. However, their focus is in the mechanics of automating policy design, not in how their complete model could in fact be architected. Their goal is to try and create a mechanism that would allow a storage administrator to automatically select the policy given the business goals and the underlying technology. Such a mechanism could be incorporated in our data management framework, automating the manual process of policy design.

The problem of automating policy design in general has been studied [1, 2, 9]. However, none of these papers present a general architecture for how such a policy can be used by administrators or software to implement the policy that is designed. The approaches presented could be incorporated into our framework to simplify the design of policies.

The use of policies and storage pools in backup applications is not new. Kaczmarski describes in TSM the use of storage pools and policies [7]. However, the intent of the policies is not to provide a mechanism to allow data administrators to mange their own data or to manage the underlying storage, but to provide TSM with more information as it makes data management decisions.

Policy-based management systems are also nothing new. Commercial systems such as Brocade StorageX, Opsware, and Symantec all utilize this approach for many management tasks. However, these policies are intended to automate tasks within an administration domain, not as a mechanism to delegate control between domains. Patterson, et al. describe the use of policies to reduce the cost of storage management by compressing or deleting old data in the SRM space [11].

Data Set Concepts

Prior to the introduction of data sets and policies, administrators drew little distinction between data and the physical containers associated with that data. As a result, administrators thought of data in terms of where they were located and how they were configured. As data management requirements changed, storage administrators had to manually map the data to the appropriate storage container and change the configuration associated with that storage container to match those new data management requirements. Managing the mapping and ensuring that the configuration is correct were time-consuming and prone to human error. As a result, configuration changes were infrequent. The net result is that storage was either over-provisioned or under-provisioned in terms of capacity, performance, or redundancy.

Data sets and resource pools do not replace the existing storage and data management containers; they are the management view of those containers. For example, a "data set" refers to the data stored by a FlexVol volume. When storage is provisioned, a FlexVol volume is actually created on the storage system; however, because their configuration is stored externally to the storage system, a data set can exist for the entire life cycle of the data, whereas a particular storage container might not. The data might move to another volume or begin sharing the volume with data in another data set. We expect that, over time, administrators will tend to refer to data sets instead of the underlying storage containers.

Resource Pools

A resource pool contains storage from which data sets can be provisioned. A resource pool can be constructed from either, disks, volumes, aggregates or entire storage systems. If a resource pool is constructed using a storage system then we implicitly add all of the disks and pre-created aggregates on that storage system, including any additional aggregates or disks that are later added. There might be many storage systems or aggregates from multiple storage systems in a single resource pool. The resource pool definition is stored in the external database.

In addition to storage capacity, a resource pool also contains the attributes and capabilities of the underlying storage systems in the pool. These attributes include the data access protocols supported, the performance of the software and storage controller, the reliability of the controller, and the data management software features that are available. These properties are automatically discovered and recorded when storage or storage systems are added to resource pools.

The conformance engine uses the capacity and attributes of resource pools when determining the best location for data to be provisioned. A single resource pool might contain different tiers of storage representing different performance and cost attributes.

A resource pool serves two purposes. The first is to reduce the total number of distinct objects a storage administrator must manage, and the second is to allow greater opportunities for space and load optimizations.

Policies such as thin provisioning limits and RBAC can be applied to whole resource pools, rather than to individual storage systems or components.

Since a resource pool consists of discrete quantities of storage, the larger the pool, the more opportunities to optimize for space and load balancing exist within that resource pool.

User-Defined Properties

Although many properties of resource pool members are discovered automatically, certain properties might be explicitly defined by administrators. This permits more flexibility and control when matching provisioning requests with available resources. For instance, it might be desirable for administrators to add an explicit property related to geography to a resource pool member. This property then might be specified as part of a provisioning policy and matched against available resources in a resource pool that has been assigned this property.

Data Sets

Data sets represent a collection of data and their replicas. In our current implementation, "data set" refers to the data contained within one or more storage containers, not the storage containers. The storage containers used by a data set might change, over time, due to load- or space-balancing actions or policy changes. These changes should be transparent to the users of the data. A data set is provisioned from existing storage containers or from a resource pool according to policy. The data set definition is stored in the external database.

Data sets also have provisioning and data protection policies. The policies apply to all of the data in the data set.

A data set serves four purposes. The first is to allow data administrators to manage their data without requiring that they understand how the storage is configured or where it is physically located. Once the data set has been defined, the administrators only have to choose the policies that best match their data management requirements.

The second purpose of a data set is to reduce the number of managed objects. A data administrator might have a lot of data that must be monitored and managed as a single unit spread across many distinct storage containers, such as an Oracle databases or Exchange applications. A data set allows both the storage and data administrators to manage and view the data as a single unit.

The third purpose is that a data set provides a convenient handle to all of the replicas, allowing administrators to view or restore versions of their data without requiring knowledge of where those versions are (or were) stored.

The fourth is that the data set provides the mapping between the physical storage and the desired behavior associated with its policies. As new storage capabilities are added to the system, or policies are changed, the data management framework can reconfigure the existing storage containers, or possibly migrate data to new storage containers, to better satisfy the data set policy requirements.

Policies

A policy describes how a data set should be configured. This configuration specifies both how the data set should be provisioned and how it should be protected. In our framework, we treat these as distinct policies.

A provisioning policy consists of a set of attributes that the data set requires from a particular resource pool. Specific attributes include – but are not limited to – cost, performance, availability, how the data can be accessed, and what to do in out-of-space situations.

A data protection policy is a graph in which the nodes represent how storage must be configured on resource pools and the edges describe how the data must be replicated between resource pools. The nodes have attributes such as the retention periods for Snapshot copies. The edges have attributes such as lag threshold or whether mirroring or full backup replication is required.

Policies are used by storage administrators to describe how storage should be provisioned and protected. These policies are then used any time data sets are provisioned, eliminating configuration errors that result from ad-hoc processes. These policies can also be given access control by the storage administrator, such that not all resources and configurations are available to all data administrators. For instance, some data administrators might not have access to policies that require highly reliable or high-performance storage (because of the expense required to satisfy those policies).

Data administrators are free to select any authorized provisioning and protection policies that meet their desired data behavior without regard to how the storage is configured or located.

In practice, the data administrator might assign provisioning or protection policies to data sets currently stored in containers which are incompatible with the policy.

For example, if a data set includes data which resides on a storage system without a SnapMirror license, a policy specifying a mirror relationship between the primary and secondary node cannot be conformed to without reconfiguration. Similarly, if a data set has a policy attached, data administrators might add members to the primary node of the data set which are incompatible with the policy. In both cases, the conformance engine can detect the conflict and explain to the data administrator why the underlying storage needs to be reconfigured or the data migrated. The administrator can cancel the operation or approve the tasks proposed by the conformance engine to bring the data set into conformance.

Conformance Engine

The conformance engine uses the policy to configure the underlying storage. The conformance engine service ensures that the resources used by the data set conform to the attributes described in the associated policy. Much of the automation associated with our management framework is derived from this construct. The conformance engine first monitors the physical environment and then compares the physical environment to the desired configuration specified by the policies. If there is a deviation from configured policy, the software alerts administrators to the violation and, in some cases, automatically corrects the problem.

The conformance engine uses the management object definitions stored in the database when it checks for policy deviations.

The conformance engine is split into two parts. This first part performs a comparison between the policy associated with the dataset and the physical environment, and then prepares a list of proposed actions to bring the data set into conformance. The second part executes the resulting actions.

By breaking this module into two parts, we allow the software to “dry run” any policy changes or requests in order to observe and confirm the resulting actions. This is an important feature in cases in which policy changes might result in highly disruptive actions, such as reestablishing a baseline replica on different storage, or in cases in which administrators simply want to review any changes before committing to them.

For example, the conformance engine periodically compares the data protection relationships protecting the members of a data set with the policy associated with the data set. For each node in the data set, the conformance engine determines the type of data protection relationship, or relationships, which policy dictates this node should be the source of. Then, for each member of the node, the conformance engine attempts to find a data protection relationship of this type originating with the member and terminating with a member of the destination node for each of the node’s outgoing connections. If a relationship is not found, a task is generated to construct one.

The results of the conformance run include the task list and an explanation of what actions the system will perform to bring the data set into conformance with the configured policy. For most actions, these tasks are executed automatically, but some require user confirmation. Over time, the set of tasks administrators are willing to automate without user intervention will grow. It is also possible that “unresolvable” tasks will be generated which require user intervention before they can be executed.

Role Based Access Control (RBAC)

RBAC controls management access to all of the management objects. An RBAC service allows a security administrator to configure which role can perform which operations on which objects. Whenever any operation is attempted on any management object, the RBAC service is consulted for authorization. The RBAC configuration is maintained in the same database as the data sets, policies and resource pools.

The RBAC system allows storage architects to delegate responsibility to data administrators. RBAC allows a storage administrator to safely allow data administrators to select policies and resource pools for their data sets without relinquishing control over the particular resources used.

Once storage resources are in use by a data set or assigned to a resource pool, data administrators, who are not permitted to manage storage directly, are able to indirectly manage them by performing operations on the data set or resource pool.

Comparing the Traditional and Data Set Views

Consider a data center with users that have home directories that are accessed with NFS and CIFS, Oracle deployments over NFS, a Microsoft Exchange deployment and varying degrees of data protection.

In Figure 2 we show the traditional view. In Figure 3 we show the data set view. The traditional view presents a detailed schematic of how the infrastructure is currently configured. The fact that there are several data sets and what the data protection and provisioning polices associated with those data sets are is obscured. Furthermore, the schematic presents a lot of information that is not always important and also hides the fact that some of the differences are irrelevant. The data set view, on the other hand, makes it very clear which data is using the infrastructure and what the provisioning and data protection policies are.

```
taskList GetConformanceTasks(dataSet ds) {
    taskList tl;
    policy p = ds->getPolicy();
    foreach (node n in p->getNodes()) {
        foreach (connection c in n->getOutgoingConnections()) {
            destNode dn = c->getDestNode();
            if (c->isBackupRelationship()) {
                 // backups are performed on qtrees
                foreach (qtree q in n->getQtrees()) {
                     bool conformant = FALSE;
                    foreach (destMember dm = q->getDestMembers()) {
                        if (dn->hasMember(dm)) {
                            conformant = TRUE;
                            break;
                        }
                    }
                    if (!conformant) {
                        // generate a task to create a backup relationship
                        tl->addTask(q, c, dn);
                    }
                }
            }
            if (c->isMirrorRelationship()) {
                 // volumes are mirrored
                foreach (volume v in n->getVolumes()) {
                     bool conformant = FALSE;
                    foreach (destMember dm = v->getDestMembers()) {
                        if (dn->hasMember(dm)) {
                            conformant = TRUE;
                            break;
                        }
                    }
                    if (!conformant) {
                        // generate a task to create a mirror relationship
                        tl->addTask(v, c, dn);
                    }
                }
            }
        }
    }
    return tl;
}
```

Figure 1: Algorithm used to compare data protection relationships against a configured data protection policy and generate conformance tasks. If no tasks are returned, the data set is in conformance.

The data set view also aggregates the physical infrastructure. Rather than seeing a schematic layout with specific components, we see that there are three different tiers of storage that can be used for protection and provisioning. The data set view is easier to use when performing daily management. The schematic view is important when you need to actually drill down to and manipulate the physical components directly, such as when the hardware breaks or needs to be changed.

Implementation

Protection Manager

The first system to incorporate the framework is Protection Manager, from Network Appliance. It allows backup administrators and storage architects to coordinate their efforts to construct end-to-end data protection solutions using a workflow-based graphical user interface. In addition to integrating several Network Appliance data protection technologies, Protection Manager uses higher-level abstractions to leverage administrators' time.

Protection Manager builds on the functionality of DataFabric Manager by extending the client/server architecture. Figure 4 shows how Protection Manager and various components that are used to protect storage relate. The dashed lines represent open network APIs.

Storage architects use Protection Manager to define data protection policies and define resource pools consisting of aggregates available for secondary storage

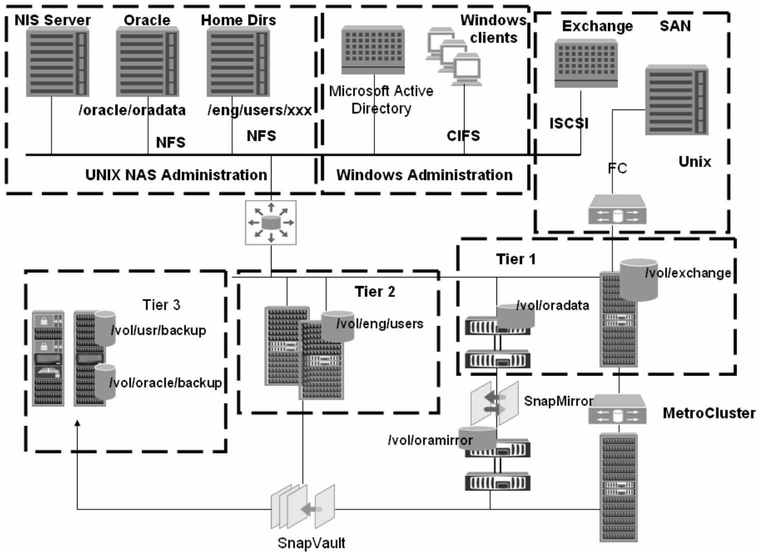


Figure 2: The storage infrastructure view presents a detailed schematic view that obscures how the data uses the infrastructure and omits the data's requirements.

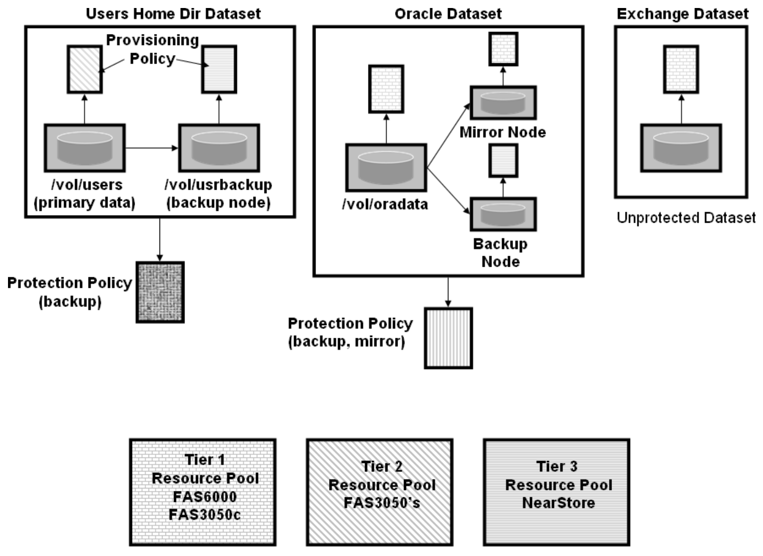

Users Home Dir Dataset
Provisioning Policy
/vol/users (primary data)
/vol/usrbackup (backup node)
Protection Policy (backup)
Oracle Dataset
Mirror Node
/vol/oradata
Backup Node
Protection Policy (backup, mirror)
Exchange Dataset
Unprotected Dataset
Tier 1 Resource Pool FAS6000 FAS3050c
Tier 2 Resource Pool FAS3050's
Tier 3 Resource Pool NearStore

Figure 3: The data set view shows what data is using the infrastructure and what requirements are associated with the data. The data set view abstracts out the details of the underlying storage infrastructure.

provisioning. Then, backup administrators define data sets by adding primary storage objects, such as volumes and qtrees. Based on the data protection policy assigned to the data set (either by the storage architect or by the backup administrator), Protection Manager provisions the required secondary storage from the configured resource pools and creates backup and mirror relationships.

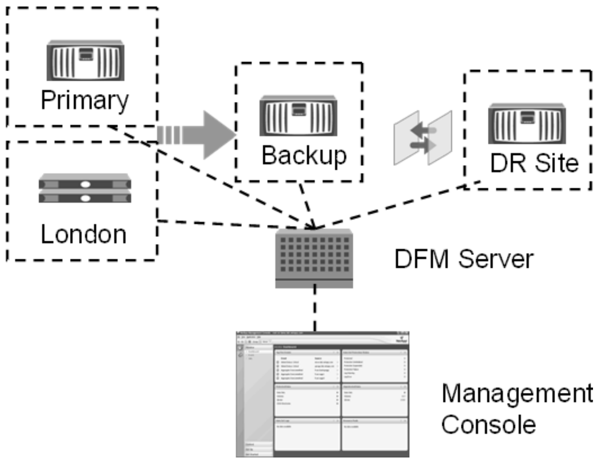


Figure 4: High-level architecture of Protection Manager.

Once the data set is in conformance with its configured data protection policy, Protection Manager continues to monitor its status for deviation and takes action to bring the data set back into conformance. Besides infrastructure errors, out-of-space conditions and policy changes, Protection Manager also monitors changes in the primary data, such as the creation of a new FlexVol volume.

As an example, if the backup administrator creates a data set, "Branch Office," and assigns the policy, "Back up, then mirror," Protection Manager knows that the administrator wants the primary data added to the data set to be protected with a local Snapshot schedule, backed up to secondary storage, and then mirrored to tertiary storage. By assigning resource pools for secondary and tertiary storage provisioning, the administrator tells Protection Manager from where to provision storage. Protection Manager will use the policy's configured schedules for local Snapshot, SnapVault between primary and secondary, and SnapMirror updates between the secondary and tertiary to protect the primary data.

If a new volume is created in a storage system and added to the data set, Protection Manager will discover the new volume and declare that the data set is out of conformance with its configured data protection policy. To rectify the situation, Protection Manager will begin making Snapshot copies of the new volume, provisioning a secondary volume, creating a SnapVault relationship between the new volume and the provisioned secondary volume, provisioning a tertiary volume, and creating a SnapMirror relationship between the secondary and tertiary volumes. Only then is the data set considered in conformance once again. None of these tasks requires any user interaction with the system.

Geography

Resource pools can be used to group storage resources by performance, availability, owner or any combination thereof, but one of the most interesting uses of resource pools is for geographic grouping. The data administrator knows the geographic location of the primary data in his data set and the configuration of resource pools by the storage administrators gives him visibility into the geographic location of potential backup and mirror sites. This gives him the ability to assign a data protection policy to his data set and select remote storage resources to implement the policy.

Availability of Services

Although the definitions of data sets, resource pools, and policies are maintained in a database within DataFabric Manager in our current implementation, the enforcement mechanism of these policies is configured on storage systems when possible. For instance, the scheduling policy for replication resides in the policy definitions, but the actual scheduling of replication updates should be configured on the associated storage systems. This approach minimizes disruptions in service should a centralized server, such as DataFabric Manager, fail.

The DataFabric Manager server itself can be configured to run in a high availability configuration such as Microsoft Cluster Server.

Impact

In this section, we first evaluate to what extent Protection Manager, using our data management framework, enables division of labor between administrators, empowers data administrators to configure storage and to what extent storage is automatically reconfigured when data protection policy changes. We then explore whether our data management framework, as implemented in Protection Manager, actually simplifies administration.

Division of Labor Between Administrators

Storage administrators and data administrators have different organizational roles. In order to successfully configure data protection, each administrator must coordinate his efforts with the other. Only the data administrator can identify the subset of stored data which constitutes a particular data set and only the storage administrator can identify pools of resources for the data administrator to use.

In a traditional environment, the data administrator must construct a request describing the storage containers he wants protected and either the degree of protection required or enough information about the use of the data set so that the storage administrator can determine the degree of protection required. This communication could take the form of a help desk ticket or an email. Either way, it must be processed manually and potential for human error is high. Furthermore, the cost of the interaction discourages frequent changes to requirements and encourages "over provisioning" whereby data administrators request more resources than they need to avoid additional requests later.

With the Protection Manager tool, the storage administrator defines resource pools and data protection policies and delegates access to them to the data administrator. Later, the data administrator defines a data set and assigns the appropriate data protection policy. The data administrator only has to know which policy is required for the class of data in his data set. In the definition of the policy, the storage administrator has already encapsulated what that means in terms of schedules, backup and mirror relationships and retention times.

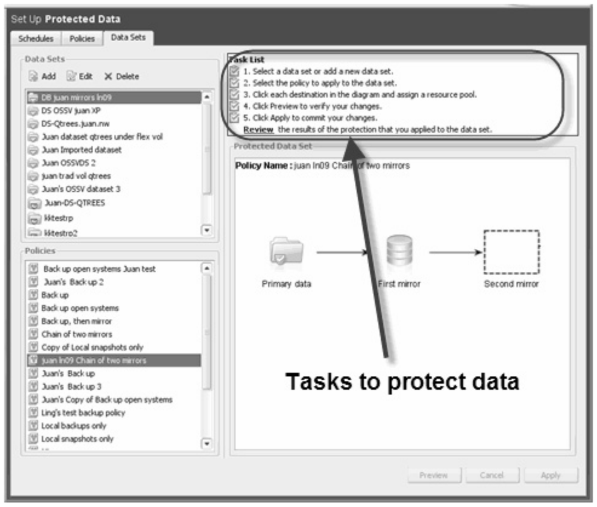
By eliminating the need to coordinate the efforts of two administrators every time a data set is protected, the Protection Manager tool simplifies the data protection process.

Empowering Data Administrators to Configure Storage

In this section we evaluate to what extent the data management framework implemented in Protection Manager actually empowers the data administrator to configure storage to match data protection requirements. To perform the evaluation, we focus on the following task: given a data set, which steps does the data administrator have to execute to configure the data set with local Snapshot copies, remote copies of a subset of the Snapshot copies, and mirrors of the remote copies of the Snapshots copies.

Before we describe what the data administrator does using Protection Manager, consider how the storage is traditionally configured. The primary volume must be configured with a Snapshot schedule. On the secondary system, a volume must be created along with an appropriate SnapVault schedule. Finally, on the disaster recovery storage system, a volume must be created along with a SnapMirror configuration that copies the data from the secondary volume. To perform these tasks without Protection Manager, the data administrator must have administrative access to the storage systems and understand the details of how to correctly configure each element.

Using the Protection Manager UI pictured in Figure 5, the data administrator first selects a policy that has been created by the storage administrator. The data administrator must then select resource pools that have been created by the storage administrator for the various nodes in the policy. At this point the conformance engine will create the secondary and disaster recovery volumes, configure the Snapshot schedule, set up the SnapVault relationship and configure the SnapMirror relationship using the APIs available on the storage system.



Task List
1. Select a data set or add a new data set.
2. Select the policy to apply to the data set.
3. Click each destination in the diagram and assign a resource pool.
4. Click Preview to verify your changes.
5. Click Apply to commit your changes.
Review the results of the protection that you applied to the data set.

DB juan mirrors ln09
DS OSSV juan XP
DS-Qtrees.juan.nw
Juan dataset qtrees under flex vol
Juan Imported dataset
Juan OSSVDS 2
juan trad vol qtrees
Juan's OSSV dataset 3
Juan-DS-QTREES
kktestrp
kktestro2

Back up open systems Juan test
Juan's Back up 2
Back up
Back up open systems
Back up, then mirror
Chain of two mirrors
Copy of Local snapshots only
juan ln09 Chain of two mirrors
Juan's Back up
Juan's Back up 3
Juan's Copy of Back up open systems
Ling's test backup policy
Local backups only
Local snapshots only

Policy Name : juan ln09 Chain of two mirrors

Figure 5: Protection Manager UI.

We have thus shown how, using Protection Manager and our data management framework; a data administrator can configure the underlying storage to meet his or her data protection requirements.

Automatic Reconfiguration of the Storage

In this section we evaluate to what extent Protection Manager allows a storage administrator to reconfigure storage automatically as a result of a change of policy. We assume that the storage administrator has two data protection policies in his or her environment. The first data protection policy only has a local Snapshot schedule. The second data protection schedule has a local Snapshot schedule and a remote backup schedule using SnapVault. To perform the evaluation we focus on the following task: changing the remote backup schedule for the second policy.

Before we describe the steps using the Protection Manager UI, we describe how the storage must be reconfigured. All of the secondary volumes that have SnapVault relationships must have their schedules changed to conform to the new schedule. To perform this task, the storage administrator must log into every secondary and modify each schedule.

Using the Protection Manager UI, the data administrator selects the policy to be modified. The storage administrator then modifies any attributes, for example, how frequently Snapshot copies should be made. The storage administrator then confirms those changes. When the conformance engine runs, it will identify the secondary volumes that must use the new schedule and modify the schedule.

Simplifying Data Management

In this section we evaluate whether the data management framework actually simplifies data management. We consider three metrics. The first is how many management entities must be administered. The second is how many tasks need to be performed. The third is whether we enable tasks that were impractical before we implemented this new framework.

The data management framework significantly reduces the number of management entities for the data and storage administrator. Rather than having to administer each individual element of a data set (for example, all of the home directories), the data administrator manages a single data set. For the storage administrator, data sets eliminate the need to monitor individual storage containers and relationships to determine whether or not the infrastructure is broken. In addition, the use of resource pools eliminates the need to manually manage space.

The data management framework clearly reduces the number of steps to perform any task. An important consequence of the automation that the conformance engine performs is the elimination of operator errors. This significantly simplifies administration.

The data management framework does enable conformance monitoring, a task that was very difficult if not impractical in large environments. Consider what it would mean to monitor conformance without the framework. Conformance has three elements. The first is a description of the desired behavior. The second is a monitoring of the actual behavior. The third is a comparison of the two. Because the desired behavior is encoded in a way that allows for comparison by software, we are able to perform this task trivially. Without that encoding, a human would have to manually compare the desired behavior to the actual behavior. Furthermore, because a human would be involved, the number of attributes that can be compared would be limited. Protection Manager's conformance engine can check any number of attributes of a data protection policy and can do it more quickly and frequently than a human could.

Conclusions

In this section, we have shown that Protection Manager allows data administrators to configure all aspects of the storage system for data protection. We have shown that Protection Manager can reconfigure the storage if the storage is out of conformance. Finally, we have shown how Protection Manager, using our framework, simplifies data management by reducing the number of managed entities, reducing the number of steps that need to be performed and enabling tasks that were too difficult to be performed manually.

Summary

In the traditional approach to storage and data management, policy is separated from the actual software that implements the policy. As a result, policy has to be carefully translated into operations by storage administrators. Each step described could introduce a human error, requiring significant time and effort to correct. By encoding the policy and being able to manage both the backup and storage process, using the data set, policy and resource pool abstractions, the storage administrator is able to perform tasks in substantially fewer steps.

Future Work

The task of modeling both cost and performance attributes in policies remains the subject of future work. At this point we have deferred to ad-hoc methods, such as manual classification of systems based on user-defined properties, to guide the provisioning process. Future versions of software will do more automatic classification of resources in this area.

Data set hierarchies are also the subject of future work, especially considering complex application architectures such as SAP or Oracle deployments. These systems place different storage requirements on different subcomponents of each application, which results in different policies. A data set representing the entire application makes sense for performance monitoring or accounting purposes, but the sub-components might be composed of separate data sets, each with unique provisioning and protection policies.

In addition, the migration of data from one storage container to another as a result of space or load balancing has not been explored.

Finally, we intend to explore further refinements of how policies are expressed. The current protection policies are topologies of how data is to be made redundant. We want to explore how administrators can describe the amount of redundancy desired and let the system determine the protection topology.

Conclusions

By introducing the concept of data sets to Network Appliance management software, we have abstracted the management of physical storage containers from the data. As a result, we can now use software to manage the data and adapt to changes in policies or resources automatically, thus making it possible for administrators to manage more data.

Author Biographies

Jim Holl is a graduate of the University of California at Berkeley with a B.S. in Electrical Engineering and Computer Science. He has worked on a variety of software platforms including Solaris and Java at Sun Microsystems and the Universal Agent Platform at Arula Systems. He currently works on storage management software at Network Appliance. Jim can be reached at jimholl@netapp.com .

Kostadis Roussos is a graduate of Brown University with a Sc.B in Computer Science and Stanford University with a MS in Computer Science. He spent the first part of his career writing a thread scheduler at SGI, and then writing a packet scheduler at NetApp. He recently made a major context switch to start

working on simplifying the management of data using storage virtualization. He can be reached at kostadis@netapp.com.

Jim Voll graduated from the University of California at Santa Barbara with a B.S. degree in computer science. He has worked on a variety of system software including compilers, file systems and UNIX operating systems. He has been working at Network Appliance as a Technical Director and is currently focused on storage management software. Jim can be reached at jjv@netapp.com.

Bibliography

[1] Alvarez, G., E. Borowsky, S. Go, T. Romer, R. Becker-Szendy, R. Golding, A. Merchant, M. Spasojevic, A. Veitch and J. Wilkes, "MINERVA: an automated resource provisioning tool for large-scale storage systems," *ACM Transactions on Computer Systems*, Vol. 19, Num. 4, pp. 483-518, 2001.

[2] Anderson, E., S. Spence, R. Swaminanthan, M. Kallahalla, Q. Wang, "Quickly Finding Near-Optimal Storage Designs," *ACM Transactions on Computer Systems*, Vol. 23, Num. 4, pp. 337-374, 2005.

[3] Edwards, J., R. Fair, E Hamilton, A. Kahn, A. Prakash, E. Zayas, "Flexible Volumes in Data ONTAP," *Network Appliance Technical Journal*, Vol. 2, Num. 2, 2005.

[4] Fore, A., J. Lyons, K. Trahan, "Beyond Backup: Disk-to-Disk Backup Comes of Age," *WP-7020-0607*, 2007.

[5] Ferraiolo, D. F. and D. R. Kuhn, "Role-Based Access Control," *15th NIST-NCSC National Computer Security Conference*, pp. 554-563, 1992.

[6] Gelb, J. P., "System-Managed Storage," *IBM Systems Journal*, Vol. 28, Num. 1, pp. 77-103, 1989.

[7] Kaczmarski, M., T. Jiang, A. Pease, "Beyond Backup Storage Management," *IBM Systems Journal*, Vol. 42, Num. 2, pp. 322-337, 2003.

[8] Keeton, K. and J. Wilkes, "Automating Data Dependability," *Proceedings of the 10th Workshop on ACM SIGOPS European Workshop: Beyond the PC*, pp. 93-100, Saint-Emilion, France, EW10, ACM Press, New York, NY, July, 2002.

[9] Keeton, K., C. Santos, D. Beyer, J. Chase, and J. Wilkes, "Designing for Disasters," *USENIX FAST 2004*.

[10] Patterson, R., S. Manley, M. Federwisch, D. Hitz, S. Kleiman, S. Owara, "SnapMirror: File-System-Based Asynchronous Mirroring for Disaster Recovery," *USENIX FAST*, 2002.

[11] Zadok, E., J. Osborn, A. Shater, C. Wright, K. Muniswamy-Reddy, J. Nieh, "Reducing Storage Management Costs via Informed User-Based Policies," *IEEE Conference on Mass Storage Systems and Technologies*, April, 2004.