Perforce Backup Strategy & Disaster Recovery at National Instruments

Steven Lysohir
National Instruments

Perforce User Conference
April 2005

Contents

Introduction

When people plan for disaster recovery, one of the first things considered is how corrupted or lost data will be restored. Another situation that frequently gets discussed is how the application server will be brought back online in the event of hardware failure. While these are very important issues that need to be addressed, many people disregard any attempt at preventing these scenarios. In reality, the first step in creating a disaster recovery plan is to look at your current architecture and processes and to identify any gaps that could lead to system failure and impede your recovery process. Your Perforce solution should be architected with not only performance as a measurable objective, but with disaster recovery in mind as well. Furthermore, your backup strategy should be designed and implemented with regards to restoring your SCM environment in the quickest and most painless way possible.

This paper details the Perforce environment at National Instruments. It illustrates how the Perforce server and application architecture, coupled with NI's backup strategy has decreased the risk of severe data loss and improved the speed and ease of recovering from a disaster. It concludes with some further recommendations to consider when developing a disaster recovery plan.

Development Environment

To provide a better understanding of the architecture and processes implemented at NI, here is a brief description of our development environment and some common Perforce metrics.

Number of Users: 1050 (globally distributed)
Number of Development Sites: 8 (globally distributed)
Number of Versioned Files on Storage Device: approximately 4.8 million
Size of Perforce Depots: approximately 600 GB
Size of Perforce Database: approximately 21 GB
Number of Changelists: just over 1 million
Number of Clients: 3508
Number of Commands Daily: approximately 100,000
Number of Administrators: One full-time administrator

Architecture

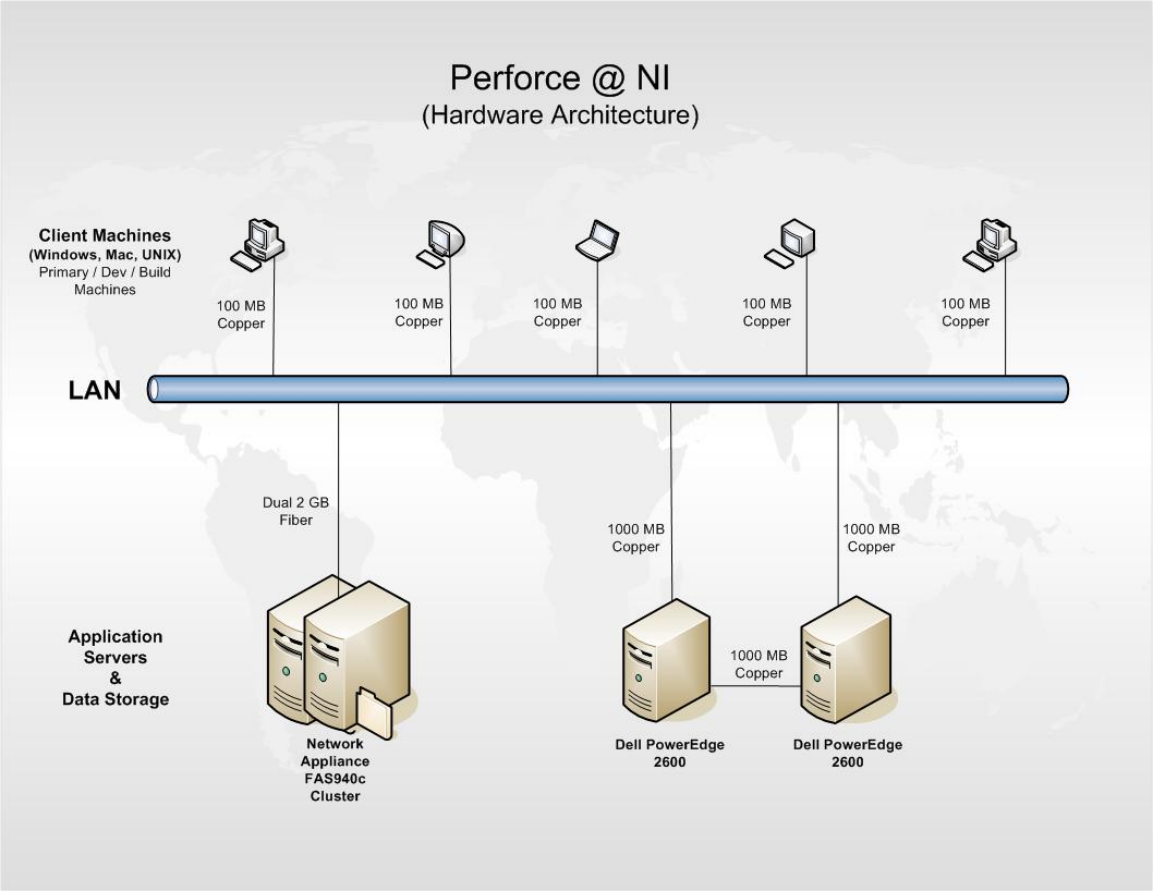
When designing a Perforce environment centered on reliability and redundancy, there were two aspects to consider: the hardware architecture and the application architecture. At National Instruments, the Perforce environment had to be architected to support the large installation described above.

Hardware

National Instruments has two identical servers in its Perforce environment. One server functions as the production Perforce application server. This server houses the production Perforce database. The second server acts as a failover server. The second server also exists as storage for older checkpoint and journal files and runs all of NI's automated, custom Perforce scripts. Both machines have identical hardware specs and are configured exactly the same.

Each server is setup with two distinct RAID arrays. These arrays do not share physical disks. The first is a two disk RAID 1 array. This array contains the operating system, program files, and the Perforce journal and log files. The second array is configured as RAID 10 with 6 disks. This array contains the Perforce application, database, and custom scripts.

Both servers are connected (via shared gigabit) to a clustered Network Attached Storage (NAS) device that functions as our main corporate file server. This device is capable of performing point-in-time copies of all data it houses. The role of this storage device in our Perforce environment is explained in the Application section below.



Application
The Perforce application has been installed following the recommendations found in Perforce's System Administrator's Guide. As recommended by Perforce, the journal and database files are stored on separate physical disks. The journal and log files are kept on the system drive (RAID 1 array) and the Perforce database resides on the data drive (RAID 10 Array). This provides added recovery capabilities in the event of disk failure. If the array containing the Perforce database is destroyed, it can be recovered by reading the journal file (which is still intact since it is stored on separate physical drives) into the most current checkpoint. This configuration theoretically provides a slight boost in performance as well, due to having the journal and database on two different spindles.

The versioned files reside on the NAS appliance on a private disk array. Although the decision to use an external storage device was driven mainly by running out of local disk space, an enterprise class storage solution was chosen based on redundancy and reliability.

There is also a copy of the production Perforce database in a separate directory on the data drive. This copy (further referred to as the "offline database") allows us to perform nightly checkpoints of the Perforce database without affecting our developers. This process is explained in the Backup Strategy section below.

Benefits
Two Servers (Production and Failover):

- Having an identical failover server allows NI to quickly switch Perforce servers in the event of hardware failure on the production server. The failover server is capable of providing the same level of service as the production server, which essentially eliminates any impact to developers. This also increases the timeframe in which the production server needs to be repaired.
- The failover server provides dedicated storage space for Perforce checkpoints and truncated journal files. This simplifies and speeds data recovery. Copying files from disk is quicker and less involved than restoring them from tape.
- Finally, the failover server provides a mirrored testing environment.

Using an Enterprise NAS Device (Storing the Versioned Files on External Hardware):

- Limits damage caused in the event of issues on the production server. The versioned files remain untouched.
- Speeds up and eases the transition of the production Perforce application to the failover server. Versioned files do not have to be copied or restored.
- The reliability and redundancy of an enterprise storage device cannot be matched by a simple application server. There is less chance of data loss or corruption to the versioned files. The availability of the versioned files increases as well.
- As an added bonus, this enabled the data array on the servers to be configured as RAID 10 since disk space was not an issue, only performance and redundancy.

Future Enhancements

- With improved file locking policies and a private connection between server and storage, the Perforce database and journal can be moved to the external storage device, joining the versioned files. This provides further reliability and reduces the resource requirements (and thus cost) of future application server purchases.
- In addition, this architecture is a step towards clustering the Perforce application to provide exceptional reliability (assuming the above enhancement is in place).

Backup Strategy

With the proper architecture in place, National Instruments next had to look at developing a backup strategy that took full advantage of the Perforce environment. This started with automating checkpoints.

Checkpoints

Creating frequent checkpoints is the most important aspect when designing a Perforce backup strategy. Versioned files can be painfully recovered from local client workspaces, but all Perforce metadata is stored in the Perforce database which is limited to one instance. If this database becomes corrupted in any way, restoring from a checkpoint is often the quickest and most reliable way to recover any lost data.

Weekly Checkpoint

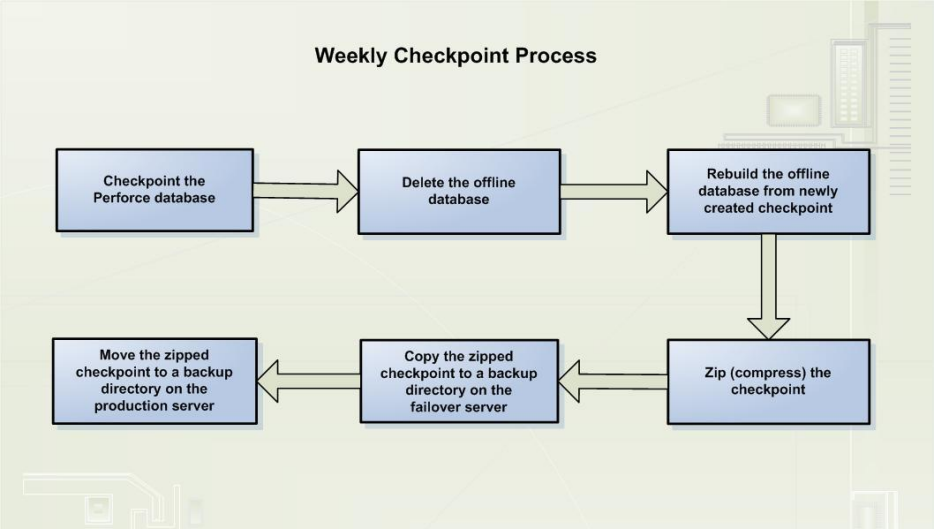
At National Instruments, checkpoints are performed on the production database weekly by an automated Perl script. This schedule was decided upon based on availability requirements for the Perforce application. Having the Perforce database locked for approximately one hour each night was not acceptable for global development.

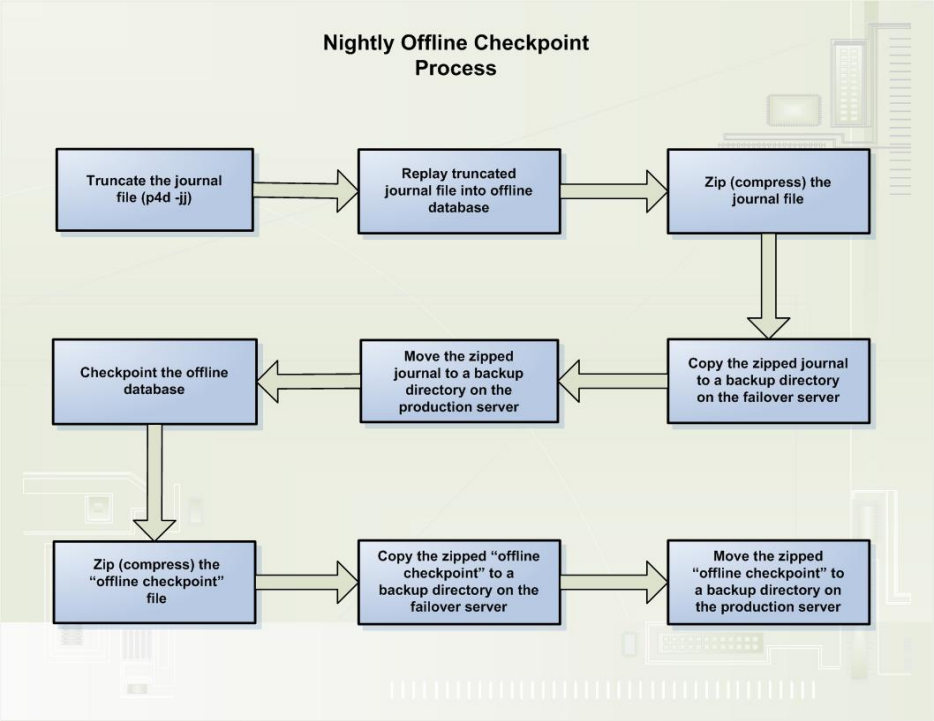
The checkpoint script begins at 1:00 AM every Sunday. Once the checkpoint is created, the offline database is deleted from the offline directory. The offline database is then rebuilt using the checkpoint that was just created. Finally, the checkpoint is zipped (compressed) and copied to a backup directory on the failover server and moved to a scripts directory on the production server.

This procedure provides a checkpoint of the Perforce database and a backup copy of the database, both in another directory and on a different server. If the production Perforce database were to become corrupt, either of the above mentioned files, along with the most current journal, can be used to restore the database to its most current, functional state. Having an offline copy of the Perforce database also allows us to perform an “offline” checkpoint nightly, that will not lock the production Perforce database.

Nightly Offline Checkpoint

As stated above, the offline database makes it possible to perform nightly checkpoints without locking the production database. This is accomplished through a Perl script that is run at 11:30 PM Sunday – Friday. First the journal file is truncated by the "p4d –jj" command. The truncated journal is then replayed into the offline database, creating a point-in-time copy of the production database. Next, the journal file is zipped and copied to a backup directory on the failover server as well as a scripts directory on the production server. The offline database is then checkpointed. The resulting "offline checkpoint" is zipped and copied following the same process as the journal file. The key benefit to this process is that you get a nightly checkpoint, backed up to another server, without any downtime to the application.





Journal Copy
While the nightly checkpoints guarantee a maximum of 24 hours of data loss, National Instruments wanted to decrease this value to 4. In order to meet this requirement, the journal file is copied every 4 hours to the failover server by a simple daemon. Each copy command overwrites the previous version so only a single journal file is present on the failover server. In the event of complete data loss on our production server, the Perforce database can be restored from the most recent checkpoint and copied journal file that exists on the failover server with only a maximum of 4 hours data loss.

Point-In-Time Copies
Another important key in the backup process at NI is the ability of the NAS device to create point-in-time copies of any data set. These "snapshots" are created instantly and consume very little disk space. At National Instruments, snapshots are created on the versioned files every 4 hours. The timing coincides with the journal copy discussed earlier. This assures that the versioned files will match the database in a disaster recovery scenario.

Enterprise Media Backups
As a final protective measure, all perforce data is backed up through an enterprise tape backup system. The only files excluded from this process are the files that make up the production Perforce database. These files are excluded to avoid locking while the backup software makes a copy to tape. However, the Perforce database does get indirectly backed up to tape through the offline database and checkpoint files that are stored on the production and failover servers.

The versioned files get backed up indirectly as well. The backup software saves the snapshot to tape instead of copying the versioned files directly. This is done for the same reasons the production Perforce database is excluded. All this data receives a full backup every night and is then shipped offsite the following day.

Benefits
Checkpoints & Journal Copy

- There is always a copy of the Perforce database readily available on disk, current to within 4 hours.
- Switching the application to the failover server is more efficient. There is no need to restore data from another source since the failover server already contains a current checkpoint and journal file. The application can be restored on the failover server simply by unzipping the checkpoint, rebuilding the database, and replaying the most current journal.

Snapshots

- The most important benefit to using snapshots is the ability to restore data directly from disk; furthermore, being able to restore data directly from disks that are part of the same array as the target. Understandably, this is pointless if the hardware was to fail or the array became compromised, but that is why there are enterprise media backups.
- Being able to perform a full backup of the versioned files nightly, without locking any files during the process. All versioned files are available to the application 100% of the time.
- Being able to create a backup copy of all versioned files every 4 hours, on the fly.

Responding To Disaster (Recovery Scenarios)

This section will attempt to illustrate the most common disasters that could occur to the Perforce environment. The benefits to the above described architecture and backup strategy will be demonstrated as you will see how they help the recovery process.

1. Disaster: Perforce Database Becomes Corrupt (Disks/Array Unaffected)
   Solution: Recover Database from Offline Database & Journal
   Process: The database can be restored by copying the offline database to the production directory and then replaying the active journal stored on the system drive.

2. Disaster: Perforce Database Corrupted Due to Disk/Array Failure
   Solution: Rebuild Array; Recover Database from Checkpoint & Journal
   Process: First rebuild the disk array. Next, copy the most current checkpoint from the failover server to the production server, and rebuild the database from the copied checkpoint. Finally, replay the active journal file safely stored on the system drive into the Perforce database.

3. Disaster: Application Server Failure
   Solution: Switch Application to Failover Server; Repair Production Server
   Process: Rename failover server to the name of the production server. Change the failover server's IP address to the production server's IP address as well. Then rebuild the database from the most current checkpoint on the failover server, and replay the copied journal file into the new database. Start the Perforce application and have users connect as usual. Finish by restoring the production server.

4. Disaster: Versioned Files Corrupted (Disks/Array Unaffected)
   Solution: Restore the Versioned Files from Snapshot
   Process: Replace the corrupted versioned files with the intact files found in the most current snapshot. This is accomplished by a simple system copy command.

5. Disaster: Versioned Files Corrupted Due to Disk/Array Failure
   Solution: Restore Versioned Files from Tape Backups

Factors to Consider When Responding to Disaster Scenarios:

- Were all files recoverable? Do any lost revisions need to be obliterated?
- Can unrecoverable files be replaced by local client versions?
- Do any developers need to be notified of unrecoverable files?
- If the production server goes down, which option is more critical?
  1. Getting the application running on the failover server? (Application quickly available despite possible, permanent data loss.)
  2. Recovering the production Perforce database? (Application will be unusable while attempting to recover all data.)

Further Recommendations

While National Instruments' Perforce architecture and backup strategy is critical to the company's ability to quickly respond to a disaster situation, there are still two more key ingredients to success: well written documentation, and clearly defined roles and lines of communication.

Documentation

Have your disaster recovery plan documented. Try to include all possible disaster scenarios with a clear, easy to follow resolution for each setting. More importantly, have the required Perforce and enterprise backup software commands documented. This information should be freely available and easy to understand. Your Perforce or backup administrator may not always be available in the event of a disaster. This documentation will allow anyone with the proper access to respond to almost any disaster situation. You never want a lack of shared knowledge to be your single point of failure.

Roles & Communication – Questions to be Answered

- Who needs to be involved in the recovery effort?
  - Perforce application failure
  - Operating system crash
  - Tape restore
  - Hardware issues
  - Communication

- Who in the business needs to be informed about the situation and how detailed should the briefing be?
  - Immediate notifications
  - Updates
  - Final resolution
  - Cause & effect analysis

- Who will communicate updates?
- Where should all end user questions be directed?
- Is everyone aware of their roles and the recovery plan?
- Should there be a post mortem and who should be involved?

One Final Note – Test Restores
It can be argued that the ability to restore your data is the most important aspect of a backup strategy (while in fact, being the whole of your disaster recovery strategy). Without the ability to perform a successful restore, the backed up data is useless. At National Instruments, a test restore of all the Perforce data is performed on a 6 month rotation to insure the process is successful and can be repeated in the event of data loss.

Conclusion

National Instruments has been very successful in protecting its Perforce environment from disaster. By implementing the above described architecture, NI has increased redundancy and reliability, thus increasing the availability of the Perforce application. In addition, the previously described backup strategy has enabled National Instruments to quickly resolve any disaster scenario by enabling virtually all data restores to originate from disk rather than tape, and by providing a preconfigured failover server with equal capacity as the production server.

While this solution works well for a large Perforce installation, it does have its drawbacks. There is an increased monetary investment in hardware and disk storage. There is also a cost in having a more complicated environment to manage and monitor. Some of these drawbacks can be eased by automation (managing the hardware and processes), and some can be justified by evaluating the consequences. Hardware may be expensive, but compare that cost to unrecoverable data loss or application downtime.

Regardless of the benefits, this solution may still not be viable to all Perforce environments. Yet the theories behind this solution can still be implemented.

- You may not have an identical failover server onsite, but you can still install the Perforce application on another piece of hardware in the office and have this hardware preconfigured to act as a temporary failover server.
- Both checkpoint processes can be followed, saving the checkpoints to any share on the network.
- To simulate snapshots, experiment with one of the many simple file copy utilities capable of performing incremental copies of data to keep two shares in sync.

The bottom line is to make your Perforce environment as redundant as possible. If that can be accomplished, you will hopefully never have to implement a disaster recovery plan.