

TECHNICAL WHITE PAPER

# Accelerate Work Area Creation for Perforce Users on FlashBlade//S with RapidFile Toolkit

Gain speed and cost efficiency for your overall software development pipeline.



# Contents

**Introduction** ..... 3

**Challenges in Software Development** ..... 3

**How to Use This Paper** ..... 4

**Perforce and Pure FlashBlade//S Architecture** ..... 5

**RapidFile Toolkit** ..... 5

**Accelerating Creation of Perforce Work Areas Using RapidFile Toolkit** ..... 6

**Test Methodology** ..... 8

**Test Results and Observations** ..... 8

**Conclusion** ..... 11

**About the Authors** ..... 11



## Introduction

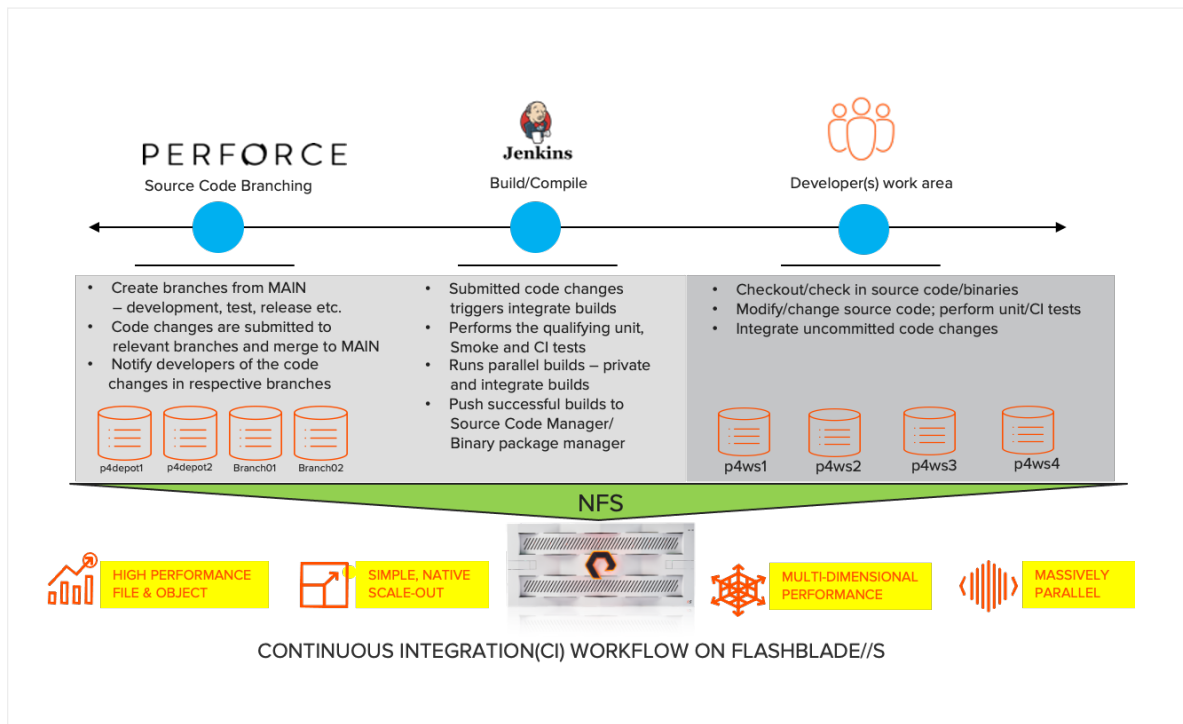
Over the years, software development processes have evolved from rigid methodologies like the waterfall model to more adaptable and iterative approaches such as continuous development (CD) and continuous integration/continuous delivery (CI/CD). These new methodologies prioritize agility and adaptability, which are critical for improving developer productivity and faster time to results. Software development teams are constantly looking for ways to optimize their processes, improve code quality, and speed innovation. By continuously improving their software development workflows, teams can ensure that they are delivering quality of results (QoR) that meet the needs of their customers and help their organizations stay ahead of the competition.

Source code management tools and binary package management systems are two essential elements in any software development life cycle. Perforce, Subversion (SVN), and various flavors of Git are common source code management tools used in various industry verticals such as electronic device automation (EDA), oil and gas, and finance.

Perforce is widely used as a source control management tool in EDA, gaming, and other organizations that handle large codebases.

## Challenges in Software Development

Although an increasing number of organizations are adopting continuous integration processes to transform their software development workflows, there are still certain challenges that need to be addressed. The primary issue is that organizations in domains such as EDA, semiconductor, media and entertainment, and gaming have huge codebases ranging from high MB to multiple GB in sizes, with numerous development branches. During checkout from respective development branches, software developers must wait to copy source code files into their work areas, change ownership and then perform a full build to complete the process.



**FIGURE 1** Sample CI workflow during the software development process

As shown in Figure 1, depending on the size of the development branches, users frequently have to check out their respective work areas and run a full build of the source code before starting to code. This process results in considerable time for onboarding developers in a large-scale setting. In the above diagram, user work areas are arranged as distinct directories and saved in a solitary file system on Pure Storage® FlashBlade//S™, which simplifies storage administration and notably accelerates developer productivity.

The Perforce commit servers undergo significant strain when developers check source code out and back in when making concurrent code modifications. This leads to a substantial amount of load in the form of high CPU usage, increased IOPS, and clogged bandwidth, which create a major performance bottleneck on the Perforce servers. Although the addition of more Perforce edge servers addresses the problem of high availability and local caching for reads, the writes (code changes) are still submitted to the primary Perforce servers.

## How to Use This Paper

The main objective of this white paper is to enable reduction of the time required to onboard developers using Perforce and improve productivity by using the [RapidFile Toolkit](#) with Pure Storage FlashBlade//S. RapidFile Toolkit is a Linux utility designed to accelerate file management and delivers two times faster results in creation of user work areas on FlashBlade//S for Perforce users.

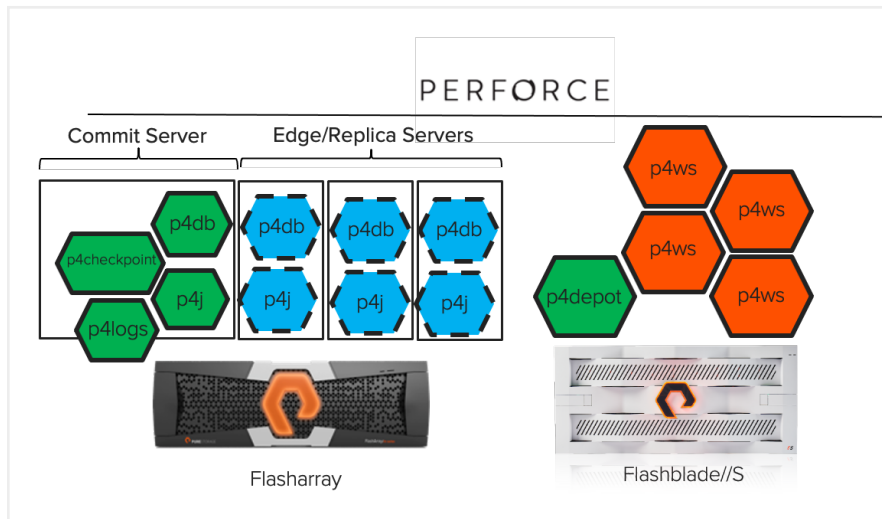
This paper is intended for project managers, DevOps engineers, and IT administrators who manage large code databases for software development projects and who specialize in Perforce deployment, configuration, and management.



## Perforce and Pure FlashBlade//S Architecture

The FlashBlade//S array is an advanced scale-out file system that supports NFS and object storage on the same platform. FlashBlade//S is a [unified fast file and object \(UFFO\)](#) platform, purpose-built to handle all the workloads generated during software development and chip design. The ability of the FlashBlade//S platform to efficiently serve all different types of I/O under high concurrency and parallelism makes it an excellent fit for high-performance environments. It enables organizations to consolidate data silos and share data in today's rapidly-evolving, data-first world.

Employing a [validated architecture](#) of Perforce on Pure Storage FlashBlade//S brings several benefits, including zero downtime for edge servers ensuring high availability, better performance for Perforce databases, scalability for user work areas, faster backup and recovery for Perforce databases, and data reduction for large repositories.



**FIGURE 2** Hybrid architecture for Perforce Helix Server with FlashArray™ and FlashBlade//S

## RapidFile Toolkit

FlashBlade//S can store extensive datasets on NFS shares with intricate directory structures and varying file sizes. Carrying out file system operations on FlashBlade//S through standard UNIX commands like "ls," "du," "chmod," "chown," "find," "rm," "cp," and "tar" can be extremely time-consuming. This is because these UNIX commands involve single-threaded serial RPC calls through a single TCP connection to the FlashBlade//S array.

Pure Storage provides a host-based utility called RapidFile Toolkit, is a Linux utility that can perform multiple-threaded operations, scaling in performance with concurrent RPC calls and multiple TCP connections to FlashBlade//S. Managing files with RapidFile Toolkit and FlashBlade//S brings significant speed-up to workflows compared to traditional Linux file utilities and storage platforms.

RapidFile Toolkit can be downloaded from the following URL (requires Pure 1 credentials):

[RapidFile Toolkit v2.1 for FlashBlade//S](#)

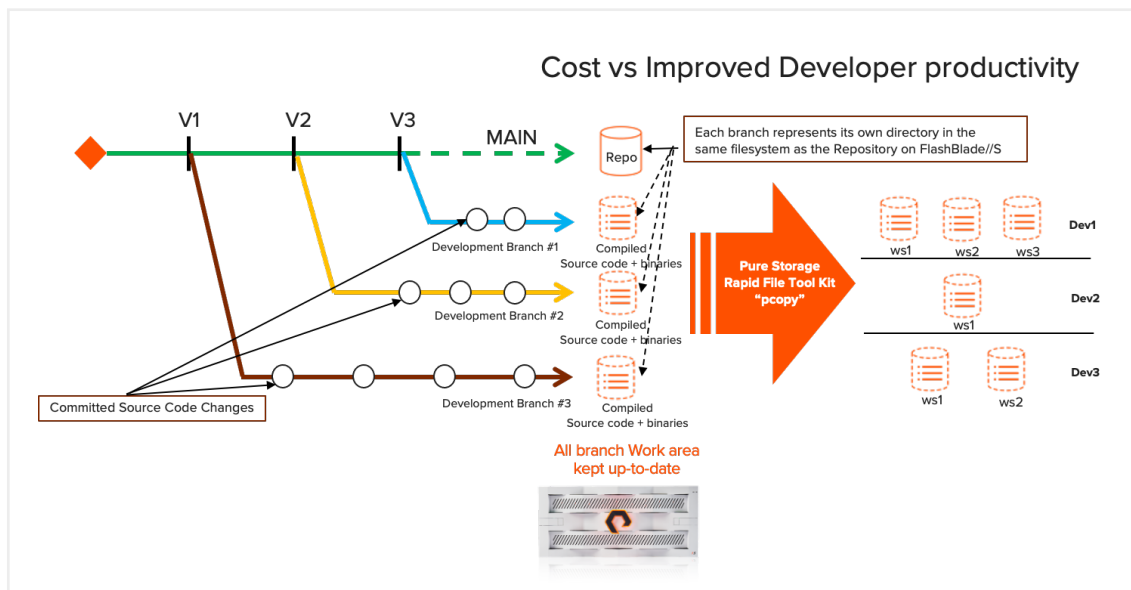
**NOTE:** The RapidFile Toolkit page on Pure Support website contains detailed information about the RapidFile Toolkit commands, and how to download and install RapidFile Toolkit.



RapidFile Toolkit includes a set of commands such as "pls," "pdu," "pchmod," "pchown," "pfind," "prm," "pcopy," and "ptar" that enable you to manage, find, and copy millions of files stored on FlashBlade//S. These commands generate parallel threads that are significantly faster (5x to 50x) than traditional UNIX commands. RapidFile Toolkit can be used in various workflows such as software development, EDA, AI/ML, genomics, and analytics.

## Accelerating Creation of Perforce Work Areas Using RapidFile Toolkit

Figure 3 illustrates the typical software development environments, which include various development branches for developers to work with.



**FIGURE 3** Sample source code branches vs. developer work areas

In a conventional Perforce production environment, the "p4 sync" command in combination with "chown", is frequently employed to duplicate all the source files, enlist the work area in the Perforce database, and modify the ownership of the files and directories. However, the "chown" process can be time-consuming, particularly for datasets larger than 100GB. This is followed by compiling the source code using "make" in the user work area before the developer can start working on the code.

Instead of pulling the code from every development branch and compiling them independently in different work areas, a more efficient method is to have each development branch with its own work area containing the source files and dependencies followed by a "full" build. The branch work area is currently populated with the source code, dependencies and the binaries.

The developers can start to code after copying the files to the user private work area from the respective branch work area. As developers commit changes to the development branch, an incremental build can be automatically triggered with the changes in the respective branch work areas before merging the changes to the MAIN code. This keeps the development branches up-to-date and notifies developers with uncommitted code changes in their private work areas. The development branch work areas can be configured in different directories, alongside the MAIN source files in the same "p4 depot" file system on FlashBlade//S.

### Accelerating the Creation of Perforce User Work Area



To accelerate creating a Perforce user work area, follow the below steps.

1. Use the RapidFile Toolkit “pcopy” command to copy the latest source code, dependencies, and binaries from the development branch area to the private work areas of the developers.
2. After the files are copied, use the “p4 sync -k” command to register the developer work area to the Perforce database.
3. As the “p4 depot” file system and the user work area are configured on FlashBlade//S over NFS, use the following “pcopy” command to copy all the source files and the binaries from the development work areas in the “p4 depot” to the private work areas of the developers.

```
pcopy -r /p4_synctest/synctest/ws50gb /p4ws_new/workspaces/wstest
```

In the above “pcopy” command, the source and the destination file systems are on the same FlashBlade//S. The source and target can also be on two separate FlashBlade//S systems. The source file system is the “p4 depot” that has the MAIN source files and the compiled development branch work areas. The target is the “p4 workspaces” file system for all the developers’ private work areas. The RapidFile Toolkit “pcopy” is capable of moving select files and directories from the source and target locations.

### Creating a Perforce Work Area Using “p4 sync” and “pcopy”

Both “p4 sync” and “pcopy” methods sync the client work area with the depot.

#### Creating Perforce user work area using the native “p4 sync”

```
[perforce@sn1-r720-a01-03 ~]$ mkdir /p4ws_new/workspaces/testwssync
[perforce@sn1-r720-a01-03 ~]$ cd /p4ws_new/workspaces/testwssync
[perforce@sn1-r720-a01-03 testwssync]$ p4 -c testwssync client -t ws50gb -o | p4 client -i
Client testwssync saved.
[perforce@sn1-r720-a01-03 testwssync]$ p4 sync
//depot/benchmark_files/00/06/353YZJ0x4SRE5akBZ60.dat#1 - added as /p4ws_new/workspaces/testwssync/00/06/353YZJ0x4SRE5akBZ60.dat
//depot/benchmark_files/00/06/4IoGHQu9Szoy0tYKDlI.txt#1 - added as /p4ws_new/workspaces/testwssync/00/06/4IoGHQu9Szoy0tYKDlI.txt
.
.
[perforce@sn1-r720-a01-03 testwssync]$ p4 clients|grep testwssync
Client testwssync 2023/04/06 root /p4ws_new/workspaces/testwssync 'Created by bruno. '
[perforce@sn1-r720-a01-03 testwssync]$
```

#### Creating Perforce user work area using Pure Storage RapidFile Toolkit “pcopy”

```
[perforce@sn1-r720-a01-03 testwssync]$ mkdir /p4ws_new/workspaces/testwspcopy
[perforce@sn1-r720-a01-03 testwssync]$ cd /p4ws_new/workspaces/testwspcopy
[perforce@sn1-r720-a01-03 testwspcopy]$ p4 -c testwspcopy client -t ws50gb -o | p4 client -i
Client testwspcopy saved.
[perforce@sn1-r720-a01-03 testwspcopy]$ pcopy -r /p4_synctest/synctest/ws50gb /p4ws_new/workspaces/testwspcopy
[perforce@sn1-r720-a01-03 testwspcopy]$ p4 sync -k
File(s) up-to-date.
[perforce@sn1-r720-a01-03 testwspcopy]$ p4 clients|grep testwspcopy
Client testwspcopy 2023/04/06 root /p4ws_new/workspaces/testwspcopy 'Created by bruno. '
[perforce@sn1-r720-a01-03 testwspcopy]$
```



## Test Methodology

The solution was validated by taking into account the following considerations:

- The P4 depot file systems were configured over NFS on FlashBlade//S arrays. P4 depot file systems refer to the file systems used by Perforce to store source code and other development artifacts.
- P4 depot file systems of different sizes (50GB, 150GB, 250 GB, and 500 GB) were created on the FlashBlade//S that were used by the main Perforce commit server for the test.
- A typical p4 work area creation process happens in three parts – “p4 sync” to read the source code from the p4depot to the newly created work area, “chown” to change the ownership, and “p4c” or “make” to compile the source code in the newly created work area.
- The tests were performed by measuring the time it took to create Perforce user work areas using the native “p4 sync” and “chown” commands, which was considered as the baseline. The time taken to compile the source code in the newly created work area was not recorded. Adding the compile time in the work area will increase the overall onboarding time for the end user.
- All the tests were repeated for the Perforce work areas using the “pcopy” command and the completion time was compared with the baseline.
- The work area creation was scaled up to 25 Perforce clients.

## Test Results and Observations

1. In pcopy method, the compiled code is stored in a work area and copied to all user work areas using pcopy. Conventionally, in Perforce, if the compile needs to be performed by a different user, the “chown” command needs to be executed. “chown” takes comparably more time than “pchown” (see the below scenario for 500gb dataset).

Time taken by “chown” command

```
[root@sn1-r720-a01-03 ws500gb]# pwd
/p4ws_new/workspaces/testws/ws500gb
[root@sn1-r720-a01-03 ws500gb]# cd ./depot/benchmark_files/34/36/
[root@sn1-r720-a01-03 36]# ls -l | head -3
total 296244
-r--r--r-- 1 root root    20669 Apr  7 01:02 0IthJkjtYa4vGM5B66g.txt
-r--r--r-- 1 root root 12074094 Apr  7 01:02 0LAuFvcs0qGxrwSsqxX.txt
[root@sn1-r720-a01-03 36]#
[root@sn1-r720-a01-03 ws500gb]# time du -sh .
500G
.

real    0m32.505s
user    0m0.400s
sys     0m4.475s
[root@sn1-r720-a01-03 ws500gb]#
[root@sn1-r720-a01-03 ws500gb]# time chown -R perforce:perforce *
real    1m24.027s
user    0m0.267s
sys     0m5.596s
[root@sn1-r720-a01-03 ws500gb]# cd ./depot/benchmark_files/34/36/
[root@sn1-r720-a01-03 36]# ls -l | head -3
total 296244
-r--r--r-- 1 perforce perforce    20669 Apr  7 01:02 0IthJkjtYa4vGM5B66g.txt
-r--r--r-- 1 perforce perforce 12074094 Apr  7 01:02 0LAuFvcs0qGxrwSsqxX.txt
[root@sn1-r720-a01-03 36]#
```





Time taken by “pchown” command:

```
[root@sn1-r720-a01-03 ws500gb]# pwd
/p4ws_new/workspaces/testws/ws500gb
[root@sn1-r720-a01-03 ws500gb]# cd ./depot/benchmark_files/34/36/
[root@sn1-r720-a01-03 36]# ls -l | head -3
total 296244
-r--r--r-- 1 root root 20669 Apr 7 01:02 0IthJkjtYa4vGM5B66g.txt
-r--r--r-- 1 root root 12074094 Apr 7 01:02 0LAuFvcs0qGxrwSsqxX.txt
[root@sn1-r720-a01-03 36]#
[root@sn1-r720-a01-03 ws500gb]# time pdu -sh .
500G .

real    0m0.240s
user    0m0.760s
sys     0m2.311s
[root@sn1-r720-a01-03 ws500gb]# time pchown -R performe:performe *
real    0m2.252s
user    0m2.392s
sys     0m5.971s
[root@sn1-r720-a01-03 ws500gb]# cd ./depot/benchmark_files/34/36/
[root@sn1-r720-a01-03 36]# ls -l | head -3
total 296244
-r--r--r-- 1 performe performe 20669 Apr 7 01:02 0IthJkjtYa4vGM5B66g.txt
-r--r--r-- 1 performe performe 12074094 Apr 7 01:02 0LAuFvcs0qGxrwSsqxX.txt
[root@sn1-r720-a01-03 36]#
```

- The test results (Figure 4) demonstrate that the “pcopy” command from RapidFile Toolkit was able to copy all the source files and binaries to the developers' private work areas in half the time (2x acceleration) compared to using “p4 sync” command from native Performe. For datasets of different sizes (50GB, 150GB, 250GB, and 500GB), the results show the same time difference in the developer work area creation process.

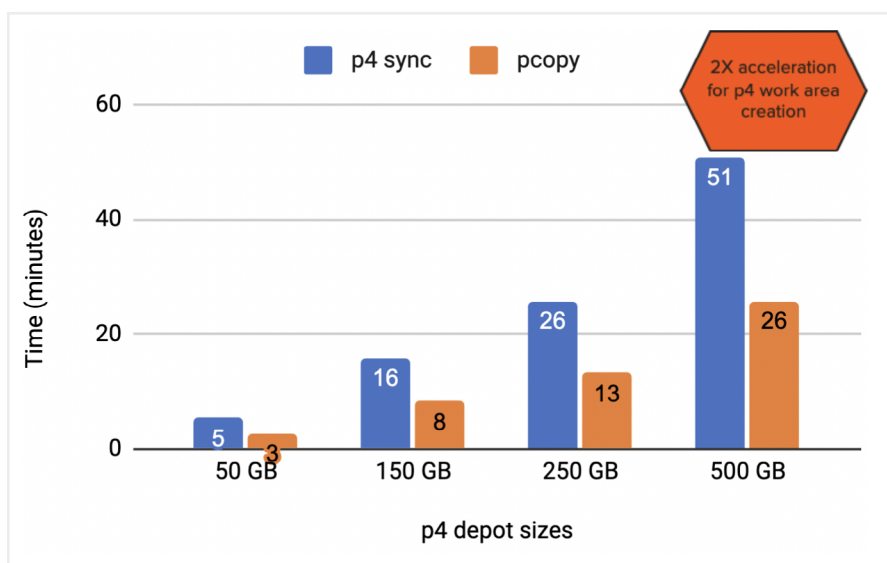


FIGURE 4 Creating Performe user work areas with “pcopy” and “p4 sync” commands



3. Another critical observation was that the Perforce server was not performing any IOPS or throughput while the “pcopy” operation was copying the files and the binaries to the developer work areas.

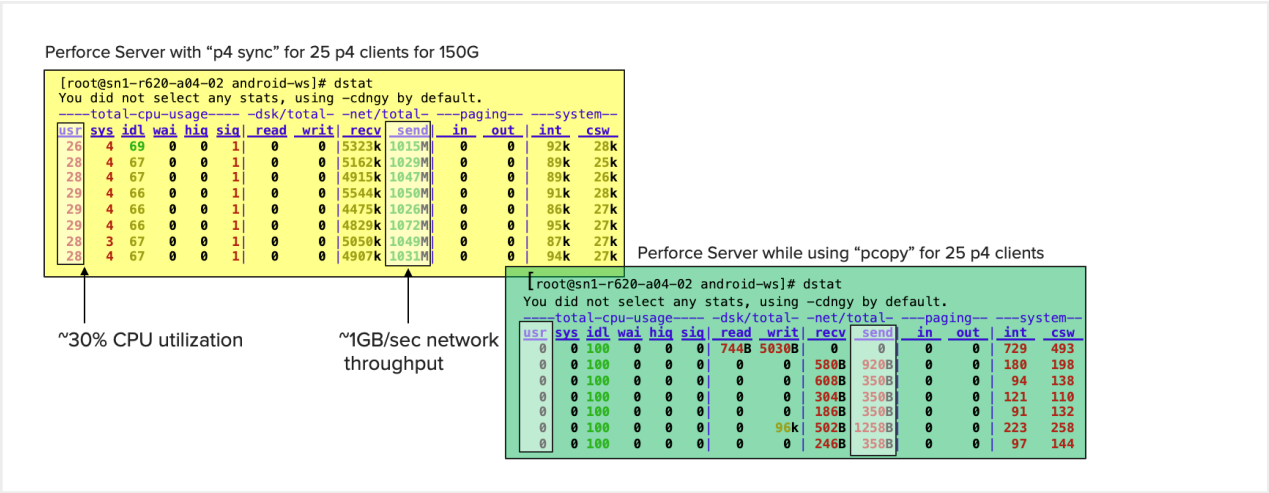


FIGURE 5 Resource usage of Perforce server when using “pcopy” and “p4 sync” commands

In Figure 5, the table on the left shows that CPU utilization for the Perforce server was approximately 30% and was pushing 1 GBps throughput when using the “p4 sync” command. A 1GBps throughput saturates a 10Gbps Ethernet connection. In most of the customer scenarios, the network becomes the most common bottleneck while scaling the number of developer work areas.

The table on the right shows that the Perforce server hardly used its resources when using “pcopy” command. Instead, it can be seen that the CPU and the network utilization were offloaded to the FlashBlade//S array. Even though FlashBlade//S array resources were heavily used during the “pcopy” operation, the duration of the “pcopy” operation was short. This allows the Perforce server to perform some core functionality to write and update committed code changes from local and remote users.

4. There was a significant data reduction on the FlashBlade//S for the user's private work areas. The test showed a data reduction of 2.6:1 for the dataset in the file system on FlashBlade//S used by all the 20 Perforce clients.



## Conclusion

This paper validates RapidFile Toolkit, a solution designed to efficiently manage millions of files is ideal for rapid acceleration of work area creation for Perforce users on Pure FlashBlade//S arrays. The solution is easy to use and set up, and requires minimal changes to existing processes and workflows. While there are many solutions available to accelerate the creation of private developer work areas, the RapidFile Toolkit "pcopy" command is unique because it can copy millions of files in parallel, quickly. This parallel copying operation allows developer work areas to exist as directories in a single file system on FlashBlade//S.

This solution allows you to offload the resource utilization overhead from the Perforce server to FlashBlade//S, resulting in cost efficiency for the overall software development pipeline. Onboarding developers is twice as fast with RapidFile Toolkit for creating private user work areas. Additionally, when you follow the methodology outlined in this paper, it can also reduce build time after code changes are submitted to the development branch work areas. Quick sync up of the source files and binaries in private work areas also boosts productivity.

## About the Authors



### Bikash Roy Choudhury

As a Director for Solutions, Bikash Roy Choudhury is responsible for designing and architecting solutions for DevOps workflows relevant across industry verticals including high tech, financial services, gaming, social media and web-based organizations. He has also worked on validating solutions with Rancher/ Kubernetes, GitLab, Jenkins, JFrog Artifactory, IBM Cloud Private and Perforce using RESTful APIs and integrating them with data platforms in private, hybrid, and public clouds. In his current role, Bikash drives integrations with strategic DevOps partners, including Rancher, Mesosphere, Perforce, GitLab and JFrog.



### Unnikrishnan R

Unnikrishnan R is a senior solutions architect at Pure Storage, with over 17 years of experience in managing IT infrastructure, enterprise storage, Cloud, and in implementing Infrastructure as Code (IaC) using Terraform and Ansible. He specializes in Linux and Kubernetes and has a passion for building robust DevOps solutions. In his role at Pure Storage, Unnikrishnan R is responsible for defining and executing complex solutions related to infrastructure and DevOps. He is well-versed in data management and migration, and has a deep understanding of data security and compliance.