Using Subversion often leads to losses in productivity and high administration costs. While its upfront licensing costs are zero, Subversion forces you to spend valuable resources on code management issues resulting from its suboptimal workflows and outdated functionality. The net loss in productivity is far more costly than licensing Perforce Software Version Management.

Perforce furnishes a compelling solution for those looking to regain productivity and resources lost to Subversion. It offers proven capabilities in an integrated platform, optimized for agile and continuous styles of product delivery. The company also offers dedicated technical support and a professional services team to ensure that your switch to Perforce is as low impact as possible.
SUMMARY

Using Subversion often leads to losses of productivity and high administration costs. While its upfront licensing costs are zero, Subversion forces you to spend valuable resources on code management issues resulting from its suboptimal workflows and outdated functionality. The net loss in productivity is far more costly than licensing Perforce Software Version Management.

Common frustrations with Subversion include:

- **Broken merging:** The incomplete merge algorithm is practically useless, requiring multiple hours to resolve the simplest issues.

- **Time-consuming release management:** The inability to track the propagation of changes to individual release branches forces long periods of code freezes.

- **Performance woes:** A large number of files in a repository or numerous revisions dramatically slow performance, driving down developer productivity.

- **Weak access control:** The coarse-grained access control makes it arduous to control where users read from and write to.

Perforce furnishes a compelling solution for those looking to regain productivity and resources lost to Subversion. It offers proven capabilities in an integrated platform, optimized for agile and continuous styles of product delivery. Perforce also offers dedicated technical support and a professional services team to ensure that your switch to Perforce is as low impact as possible.
## COMPARISON MATRIX

<table>
<thead>
<tr>
<th>Capability</th>
<th>Subversion</th>
<th>Perforce</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Merging</strong></td>
<td>Subversion has only basic merging capabilities with limited baseline selection, leading to unnecessary and incorrect merge conflicts.</td>
<td>Automatic tracking of merge history across all branches includes renames and moves.</td>
</tr>
<tr>
<td><strong>Release Management</strong></td>
<td>In SVN, users identify branches only by convention. Discovering changes that need to be applied across several branches is complex and error-prone.</td>
<td>Perforce’s Streams implementation identifies branches as mainline, development, and release with a clear hierarchy that makes it simple to identify which changes need to be propagated and to which stream.</td>
</tr>
<tr>
<td><strong>Performance and Scalability</strong></td>
<td>Struggles with large repositories and files sizes. Limited to about 250 users and 1 TB of data.</td>
<td>Proven installations of 10,000 and more users and petabytes of data.</td>
</tr>
<tr>
<td><strong>File History</strong></td>
<td>Provides only a simple log command with few filter options. A simple report to find changes of a particular user or a deleted file can take minutes because it requires listing the whole history of a project and filtering the output.</td>
<td>Supplies a strong set of tools from the command line and P4V to create a report on the file history. Tools in P4V such as TimeLapse View™, Revision Graph, or Folder Diff are essential to discover the source and history of a change.</td>
</tr>
<tr>
<td><strong>Global Teams</strong></td>
<td>No built-in caching or replication technology is available to support global workforce.</td>
<td>Perforce offers Perforce Proxy and full and filtered replication to enable fast global access.</td>
</tr>
<tr>
<td><strong>Security</strong></td>
<td>Subversion itself does not provide any security access control; it has to rely on Apache instead.</td>
<td>Perforce has built-in powerful access control and auditing facilities.</td>
</tr>
</tbody>
</table>
MERGING

Subversion
Subversion offers only basic merge history tracking, which fails in many scenarios. Such merges can be very slow and often cause unnecessary merge conflicts. This is especially true for renames and moves. Some customers report that they spend up to 5 work days in each release cycle handling Subversion merges.

Perforce
Perforce automatically tracks the history of all branch operations with its advanced and mature merge tracking mechanisms, including renames and moves. Changes can be safely merged across all branches, and the integration algorithm determines the best common base to reduce merge conflicts.

Why Should I Care?
Failed merges are the #1 concern of Subversion administrators. This often leads to the high cost and lost productivity of having users spending many hours performing the necessary merges, or for the team to abandon branching and merging altogether.

RELEASE MANAGEMENT

Subversion
In Subversion, users identify branches only by convention and there is no relationship between these branches. There is no built-in method to identify that fixes need to be applied to different branches, and recognizing the required changes is complicated and painful because of the merge implementation.

Perforce
Perforce Streams provide a guided branching framework following best practices for mainline, development, and releases. With the P4V Streams Graph, developers and release managers can readily:
- Discover which changes need to be propagated between different streams
- Perform the merge safely and quickly

These capabilities allow code line managers to easily keep track of fixes across different code lines.

Why Should I Care?
Bug fixes often need to be applied to several release branches to avoid regression. Without guidance about where these changes need to be applied, code line managers spend a lot of time tracking down individual changes by hand. In Subversion, this leads to frequent code freezes and locked-down branches.
PERFORMANCE AND SCALABILITY

Subversion
Subversion’s architecture uses forward deltas to store versions. For each checkout, it must construct the latest version from scratch by sequentially composing all individual deltas. This operation is CPU bound, as well as time consuming. Therefore, Subversion does not scale well for large numbers of users and data. Its performance suffers severely as the number of files and revisions increases and for operations that check out the latest code, such as when creating or updating a working copy.

Subversion is typically limited to about 250 users, and repositories rarely exceed 1 terabyte because of performance and maintenance problems.

Perforce
Perforce has proven installations with more than 10,000 users accessing a single repository. Many Perforce repositories hold millions of files and many terabytes of data, in some cases up to petabytes. Case studies have shown that Perforce is 5 to 10 times faster than SVN when syncing large numbers of files.

Why Should I Care?
When projects mature, their code base and history grow. If the repository cannot keep up with the increased load, the project becomes impossible to maintain. Operations that used to take seconds might now take many minutes, which will have a severe impact on developer productivity. This problem is exacerbated if you plan to connect Subversion to a continuous integration system, where the total read operations could be millions a day, mimicking 1,000 times the load created by 250 users.
Why Should I Care?
Although a traditional central server offers many benefits such as closer collaboration, there is a growing trend to be able to work while being disconnected. Sometimes the developer is on the move and unable to access the company network. More importantly, distributed version control allows developers to quickly create local branches without affecting the size of the central repository. In the end, work still needs to be stored in the central repository for sharing, reviewing, and continuous integration.

GLOBAL TEAMS

Why Should I Care?
Many development teams are going global, with users accessing and submitting to repositories from all over the world 24 hours a day. Without proper caching or replication technology, remote teams suffer from latency and bandwidth saturation, which severely hampers their productivity.
SECURITY

Subversion
There is no built-in security in a Subversion repository and no concept of access control, users, or groups. To authenticate a user and to determine whether the user has access to the repository, you must install and configure Apache in addition to Subversion, which adds extra complexity and requires additional resources. Apache access control is also typically limited to a whole repository.

Perforce
Perforce offers built-in fine-grained access control to all files for individual users and groups. Authentication is available through local password storage as well as through connectors to services such as LDAP/AD. Access to any file content can be logged for auditing.

Why Should I Care?
Your repository contains your company’s intellectual property — its most valuable asset. You need to be able to control and audit access to these assets for business, security, and regulatory reasons.

CONCLUSION
Open source products such as Subversion are not “free.” They come with hidden costs, such as lost productivity, delayed products, and increased maintenance that have the potential to be an order of magnitude more expensive than the savings in an upfront license fee.

We invite you to try Perforce for free (for up to 20 users) at perforce.com.