



# Research Report

## MKS Toolkit Vs. Cygwin: You Get What You Pay For

### *Introduction*

MKS Toolkit and open source Cygwin both do the same thing: 1) each program environment provides a Unix like environment that can run on Windows; and, 2) each program environment allows administrators to manage Unix and Windows systems using Unix commands. Accordingly, information technology (IT) managers and administrators can more easily manage Windows environments using familiar Unix constructs. Further, developers can easily migrate programs from Unix to Windows using either environment.

*Clabby Analytics* (that's me) likes both of these environments. But I also see substantial differences between the MKS and Cygwin offerings in terms of support, integration, documentation, knowledge-base maintenance, and services. In this *Research Report*, I will share my perspective on some of the differences between these environments.

*My key finding is this: because the Cygwin environment is funded only through donations, the Cygwin team does not have the kind of funding needed to provide the depth of product extensions, the level of integration, as extensive a knowledge database, as in-depth educational material, nor the extensive support that MKS provides with its products. Adopters need to understand these differences when choosing between these two offerings. If an adopter's intent is to use a more do-it-yourself environment, I would recommend Cygwin. But if an adopter wants a more integrated, tested, and better supported environment, than I'd recommend the MKS environments. Ultimately, you get what you pay for...*

### *The Biggest Difference: What Funding Enables MKS to Do...*

The biggest difference between MKS environments and Cygwin lies in the very nature of each organization's go-to-market approach. MKS charges license and support fees for its software; Cygwin accepts donations.

Because MKS charges license and support fees, MKS has a revenue stream that it can use to perform a good deal of product integration, testing and quality assurance — and that it can use for support activities (including documentation, maintaining a knowledge base, and troubleshooting). Because Cygwin does not have a similar revenue stream, it simply cannot provide the same level of integration, testing or the same level of support.

As examples of these differences, consider the following:

- The MKS Toolkit has an extensive library with over 2500 Unix commands and system calls integrated into its product offerings — including often-required "Perl" and "run" commands. For commands and utilities that have not been integrated in Cygwin's product set, Cygwin's website recommends that administrators visit <http://cygutils.fruitbat.org/>. Instructions for how to download and install these additional utilities are also found on this website.

## MKS Toolkit vs. Cygwin: You Get What You Pay For

- From a support perspective, Cygwin will attempt to fix “valid” bugs. MKS has a full support organization that is chartered with fixing level 1, 2, and 3 bugs.
- From a services perspective, Cygwin does not offer professional services. MKS, on the other hand, offers a wide variety of services to support its products, including quick start, planning and assessment, and migration services.
- MKS’s knowledge base is more extensive. And,
- MKS documentation has a more educational/learning oriented pedagogical approach than Cygwin’s approach (Cygwin’s approach is essentially “read the release notes and the user guides and have a nice day...”).

*Again, both of these products will do the job of managing Windows environments using Unix commands — as well as allow for simplified application migration. The primary difference is that because of its funding model, MKS can build a more extensive, better integrated, and better supported product than open source Cygwin can.*

### *A Closer Look: Support*

One of the big problems from a support perspective is sorting out whether the support issue is a “user problem” or if it is a “product problem”. MKS has support personnel in place that can help sort out one from another and provide the needed advice or take the appropriate action to fix a reported problem.

Note that MKS provides easy access to phone support, email/web request service, automatic product notifications, product upgrades and patches, and to its enhanced online support services portfolio.

Cygwin does not provide easy access to phone support (no support phone number is available on the Cygwin site). Further, a closer look at Cygwin’s approach shows that Cygwin developers want their users to be very careful about asking “stupid” questions.

At <http://cygwin.com/problems.html>, Cygwin developers suggest that a user’s first port-of-call when seeking support should be to visit the Cygwin FAQ and mailing list archives. If users are unable to resolve their problem, they are instructed to write to the Cygwin list to see if the community can help resolve the issue.

The Cygwin team also provides the following advice: “take a moment to read and understand some very good general advice on how to ask smart questions. Once you’ve followed that link and read the advice, please demonstrate that you’ve actually *gained some smartness* by **not** sending your Cygwin question to the authors at the link. That would be a really **stupid** thing to do.”

*As mentioned above, due to its revenue stream MKS has a well-structured support process as well as support resources charted to address problems that may arise. Because Cygwin does not have a similar revenue stream, its support policies appear to be a bit more “abrupt”.*

## MKS Toolkit vs. Cygwin: You Get What You Pay For

### *A Closer Look: Testing*

Professional grade software goes through extensive testing and quality assurance (QA) cycles. Because professional grade software generates revenue that can be used for QA, bugs and anomalies are identified in these testing cycles, and fixed by the developers of such software products.

The open source model on the other hand, does not generate the kind of revenue needed to support rigorous testing. It should be noted, however, that sometimes the open source community polices itself when it comes to QA testing (but often that community is small and testing is superficial). Cygwin cannot afford to do the level of testing and quality assurance that MKS does. And, therefore, it is logical to extrapolate that bugs will be found after Cygwin releases. And it should be noted that these bugs will be the responsibility of the user to fix (unless Cygwin agrees to fix the bugs that have been reported). And given Cygwin's stand on support, those fixes may – or may not be – forthcoming.

### *A Closer Look: Professional Services*

Cygwin does not provide professional services – period. So if you choose to adopt Cygwin, you're on your own when it comes to modifications and extensions (although adopters may be able to find some to perform professional services by contacting users on the Cygwin mailing list). By contrast, MKS has an extensive services organization, including process consulting, SCM technical consulting, and educational and training services.

MKS' Consulting Services organization offers a full range of professional services for Unix and Windows, including planning services, on-site quick-start training, and custom consulting services. And, MKS has extensive Unix-to-Windows migration expertise.

***MKS also takes-on customization work. And this is a very important differentiator when compared to Cygwin. Cygwin is open source code built and maintained by contributors. So, if the contributors take Cygwin in the direction that the user wants it developed — then great. But if a user wants or needs a new feature that is not in the plans, then that user can a) develop the products his or herself and submit it for consideration and packaging in a future Cygwin release; or b) develop the extension buy not share it. MKS, on the other hand, will perform custom development and integration work on its toolkits.***

### *A Closer Look: Integration*

The Cygwin philosophy is to turn Windows into Unix, while the MKS philosophy is to bring the power of Unix to Windows. To elaborate, the tools in Cygwin are ones that also exist in Unix. MKS, on the other hand, has over 80 additional commands that don't exist in Unix but provide Windows management from the command. These commands can be found at:

<http://www.mkssoftware.com/products/tk/commands.asp?product=win32>

The advantage of MKS approach is that it allows those users familiar with Unix to automate windows functionality using the power of Unix (typically by using command line scripts). In addition when philosophies differ between Unix and Windows (such as file permissions for example), MKS handle the difference using the Windows approach. For example, in the case of file permissions, MKS uses "chacl" and "lasacl" commands

## MKS Toolkit vs. Cygwin: You Get What You Pay For

that allow a user to modify the ACL information on not only files but registry settings and various other Windows objects. In addition there are "mkshare", "rmshare", and "lsshare" commands that allow one to create, remove and list Windows file shares, computers and network resources.

Another example is managing user and group information. Managing this information within Windows is very different than managing the same information within Unix. So MKS provides "userinfo", "groupinfo" and "member" that allow a user to write scripts to manage users and groups respectively. The way Windows handles log information is completely different as well. To handle log information, MKS provides an "eventlog" utility such that log information is accessible to a script.

Background applications known as "daemons" on Unix and "services" on Windows is another area that differs greatly between the two platforms, so MKS provides the service tool that allows the management of Windows services. A couple of concepts that don't exist on Unix are the registry and associations. MKS provides a registry tool that allows access to the registry. Using "shexec", "assoc", and "ftype" to allow a user to execute applications based on association, set file extension associations, and set file type associations respectively.

### *Summary Observations*

Both the Cygwin and MKS products will enable their users to manage across Unix/Windows environments more easily as well as port applications more easily. The primary difference between these offerings is each organization's go-to-market model. With MKS's offerings, you are paying for integration, support, and ongoing maintenance. With Cygwin, you pay nothing — so your expectations should be lowered accordingly. For do-it-yourselfers who are comfortable downloading tools and utilities and integrating them into the Cygwin base, who have low training needs — and who are comfortable relying on a community of users for support — the Cygwin approach should work just fine.

On the other hand, for users who don't have the time to figure-it-out themselves, and who don't want to spend time laboriously searching for the right utilities or the right answers to their questions, the MKS offering is a much better approach. Remember: *time is money* — and when your organization is spending its money to have developers deploy, integrate, and extend Cygwin, you are paying for that privilege.

***When you pay somebody something to integrate various utilities and commands, build an extensive knowledge, and provide educational materials and training — you get a completely different, richer product and a higher degree of support. And this, in a nutshell, is MKS's key value proposition. Ultimately — you get what you pay for...***

---

**Clabby Analytics**  
**<http://www.clabbyanalytics.com>**  
**Telephone: 001 (207) 846-0498**

© 2009 Clabby Analytics  
All rights reserved  
June, 2009

*Clabby Analytics is an independent technology research and analysis organization that specializes in information infrastructure and business process integration/management. Other research and analysis conducted by Clabby Analytics can be found at: [www.ClabbyAnalytics.com](http://www.ClabbyAnalytics.com).*