A Semi-Technical White Paper: An Introduction to Wine

codeweavers

Wine: An Introduction

Overview: Wine is a Windows compatibility technology that allows a wide variety of Windows software to run as-if-natively on Unix-based operating systems like Linux and Mac OS X. Using Wine offers the best of both worlds—being able to run the applications you want on the operating system you want. This paper discusses Wine as a technology.

Using Wine offers the best of both worlds—being able to run the applications you want on the operating system you want.

What is Wine?

Wine is a technology developed under the auspices of The Wine Project, an open-source community of developers in much the same mold as the developer community that created the Linux operating system in the first place. The Wine Project's goal is simple—to recreate the functionality of the Windows operating system and make it available on the other Unix-based operating systems. By doing so, operating systems like Linux and Mac OS X will become fully Windows-compatible platforms, allowing their users the freedom to run not only their respective native software offerings, but Windows applications as well. In the fifteen years since the project's inception, Wine has made remarkable strides towards achieving this overall goal.

What's Under the Hood?

Technically speaking, Wine is not an emulator technology. Emulators run an actual copy of the Windows operating system within a virtual machine that is hosted by the target operating system. Wine is not a virtual machine. Rather, Wine is intended to be a complete re-implementation of the Win32 Application Programming Interface (API) under Unix. It runs directly in the target operating system, and uses the native file system.

In layman's terms, re-implementing the Windows API means that a Windows software program interacts with Wine—and thereby the target operating system—directly. Wine sits between the application and the operating system. As the application makes system requests—calls for screen redraws, printing services, and so on—Wine interprets those requests and makes them

understandable to Unix. The host operating system responds, fulfilling the requests, and satisfying the needs of the application. So far as the application is concerned, it thinks it's running on Windows—it's needs are being met, so it detects no difference. has no way of telling anything's different. At a very basic level, then, Wine is simply fooling the application into thinking it's running on Windows, when it actually isn't.

One of Wine's key benefits is that it allows Windows programs to be directly integrated into the Unix operating system.

Why is this Cool?

Wine's technical approach leads to some distinct benefits compared to traditional emulation solutions. First, unlike emulator technologies—which in effect cordon off the application being run from the underlying native operating system—one of Wine's key benefits is that it allows Windows programs to be directly integrated into the Unix operating system. The Windows applications are considered normal applications of the host operating system, meaning that a Linux user sees Windows program icons on their Start Menu items, just as they would expect on a Windows machine. A Mac user, likewise, finds those same applications on their Finder.

The same benefits apply to the file system. The Windows programs directly access the host file system, allowing files to be named, saved, and retrieved just as if they were on a Windows hard drive. There's no need to move files back and forth between the host operating system and a virtual Windows partition. As a result, to the user the program operates transparently, just as their Windows application operates. No additional training is required.

Wine typically offers better system performance than an emulator running the same application.

Another key benefit regards system resources. First off, you don't have to boot or initiate Wine, as you would an emulator. Programs running under Wine are launched immediately upon being invoked. More important, of course, emulators require actually running a copy of Windows, meaning that the host system has to simultaneously "feed" two different operating systems—the host system (Mac OS X or Linux) and Windows. Both operating systems consume disk space, RAM, and processor cycles, negatively impacting their performance. Wine, on the other hand, consumes very little in the way of overhead—typically only 1-2MB of RAM per application—leaving the host operating system to perform normally. The result is that Wine typically offers better system performance than an emulator running the same application.

The final major benefit, of course, is the fact that Windows is not required in order to run Wine. Wine is a complete reimplementation of the Windows API, and therefore a complete replacement of Windows itself. Not having Windows present on the host system, in turn, protects that system from the sorts of

Not having Windows present on the host system protects that system from the sorts of malware and viruses that prey upon Windows machines.

malware and viruses that prey upon Windows machines. Unlike emulation solutions, wherein a user is well advised to install commercial anti-virus software (Macafee, Symantec, etc.) to protect them from malware when running Outlook or Internet Explorer, these same threats are not present when running Wine. Why? Because the tailored attacks that viruses use to exploit vulnerabilities within Windows are not applicable to Unix-based operating systems. In a nutshell, for all practical purposes, viruses simply don't run under Wine.

The final benefit of Wine's not requiring a copy of Windows is that you don't need a Microsoft Operating Systems license, either. This potentially frees Wine users from having to deal with Microsoft's onerous per seat licensing policies, as well as their monopolistic pricing structure.

Because of this host of benefits, Wine is a key enabling technology. It reduces the barriers to adopting Linux and Apple desktops by enabling those different platforms to run the Windows office productivity software that much of the business community has come to rely upon. As Wine matures, the barriers to adopting these alternative technologies will become lower. The end result will be more freedom for the individual consumer, as well as the re-introduction of much-needed competition into the desktop operating system space, which has been essentially strangled by Microsoft for the better part of two decades.

What are its Limitations?

Like many promising open-source technologies, Wine is still evolving at a tremendous rate. As such, it is not a finished, polished technology yet, and there remain gaps in its coverage of the Win32 API. Likewise, not all of the portions of the API that have been implemented are fully refined. Put simply, there are still lots of bugs—things don't operate quite as expected, or perhaps operate not at all. This is the necessary outcome of a technology that is attempting to recreate the product of the world's most powerful software company (Microsoft) on a budget that does not remotely replicate that of Windows' creator. So, Wine is not (yet) a silver bullet that magically allows any Windows program to run under flawlessly under Linux or Mac OS X. Typically, getting a Windows application to run cleanly under Wine still requires a degree of effort and expense.

That said, however, it is clear that Wine is moving in the direction of becoming a silver bullet sort of technology. Evidence of this lies in the steadily decreasing amount of effort required to get Windows applications to install cleanly and come up to their main screen under Wine. As late as 2000, the odds of a given application running acceptably "out of the box" under Wine were

fairly small. In 2008, those odds are probably approaching 50/50, and getting better all the time.

The substantial effort CodeWeavers has put into making the MSOffice suite run under Wine, for instance, has had the effect of making other applications run as well.

Interestingly, too, since Wine is a general-purpose reimplementation of the Win32 API, it experiences synergistic benefits from new developments across the breadth of its codebase. That is to say, a patch to Wine to fix a bug in Application A often has the side benefit of fixing similar bugs in Applications B, C, and D as well. Thus, the substantial effort CodeWeavers has put into making the MSOffice suite run under Wine, for instance, has had the effect of making other applications run as well. Thus, it is not necessary for every Windows application out there to be directly "looked at" in order to begin running. In many cases, once Wine has reached a certain critical mass, applications that heretofore never ran will "magically" begin running as well.

CodeWeavers' Relationship to the Wine Project

CodeWeavers remains the most important corporate sponsor of the Wine Project, having contributed more than 40% of all the patches in Wine (including a lot of the really hard stuff). 100% of CodeWeavers' Wine-related development work is given back to the Wine Project. And while CodeWeavers maintains a separate Wine development code tree for its CrossOver products, this tree and the free Wine tree are constantly being merged and updated based on the most promising contributions in both. In other words, unlike some other Wine-based businesses, who do not actively return their code to free Wine, CodeWeavers has fully abided not only by the letter, but also by the *spirit* of the open-source license that governs Wine's development and usage. In a sense, CrossOver can be thought of as a polished, tested, and supported version of free Wine.

CodeWeavers has fully abided not only by the letter, but also by the *spirit* of the open-source license that governs Wine's development and usage.

Conclusion

Wine remains one of the most exciting open-source technologies in the computing marketplace today. Via Wine, businesses around the globe are beginning to realize the benefits that Linux and/or Mac OS X offer, while at the same time preserving the substantial investments they may have already made in Windows software applications. Wine is thus a critical bridging technology that helps Linux and Mac users achieve the best of both worlds.