A network TEX Live installation
at the University of Groningen

Abstract
This article describes a network TEX Live installation for Windows users and the context in which it operates.

Keywords
TEX Live, MiKTEX, installers, editors, roaming profiles, Windows Vista

Our university has a LAN-based TEX installation for Windows users. The current edition is based on TEX Live. After some historical notes, I discuss the computing environment at the university, the new TEX Live-based installation and issues with Windows Vista.

This article can be considered an update of a previous article[1] about the MiKTEX-based installation that I maintained for the university's economics department.

Prehistory: 4TEX

Our department has had a network TEX installation for DOS/Windows users since the early nineties. It started out as a simple 4DOS menu for running TEX and associated programs. Later, it evolved into 4TEX, and was also made available to members of the Dutch-speaking TEX users group (the NTG) and others; see figure 1.

The final version was a Windows program, based on the same core as TEX Live. It included a vast array of utilities, some of them third-party or shareware.

A MiKTEX-based installation

When I took over in 2003, 4TEX was no longer being developed. We chose to make a fresh start, based on what was then available, and to limit ourselves to free software.

Modern LATEX editors such as TEXnicCenter can take care of running BIBTEX and MakeIndex, include spell checkers, and offer help in entering LATEX macros and in debugging. For many users, the editor is all they see from the TEX installation.

The good integration of MiKTEX as the TEX implementation and TEXnicCenter as editor and frontend was hard to argue with, so that was what we used.

For graphics support, there was a script to install WMF2EPS and its PostScript printer driver which did the real work in the background. For anything else, users had to look elsewhere.

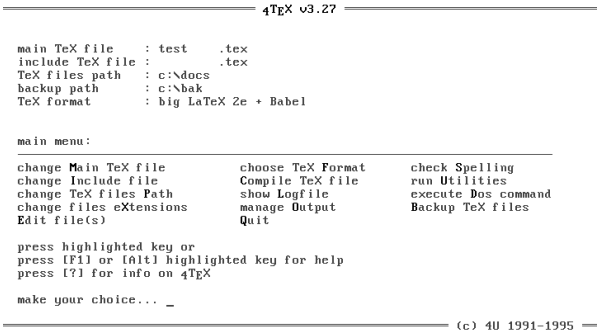


Figure 1. 4TEX main menu

Evolution

Installer. The original installer was a combined batch-file/Perl script, which also used some registry patches. This was replaced with a GUI installer based on NSIS, an open source installation system. Where possible, files and registry settings were generated during installation rather than copied from a prototype installation.

Updated versions. The editor and MiKTEX have been updated several times. This included a major change

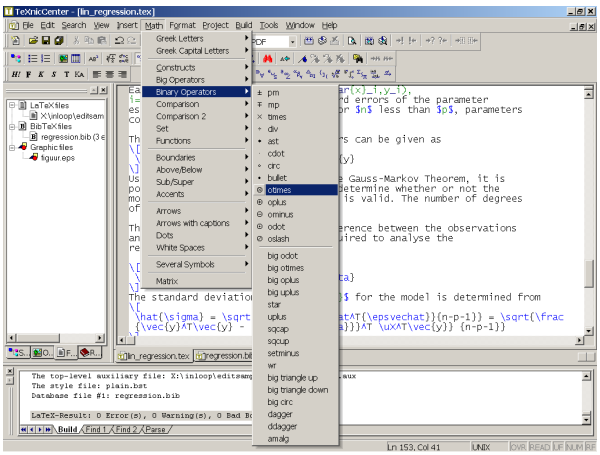


Figure 2. TeXnicCenter as frontend to MiKTEX

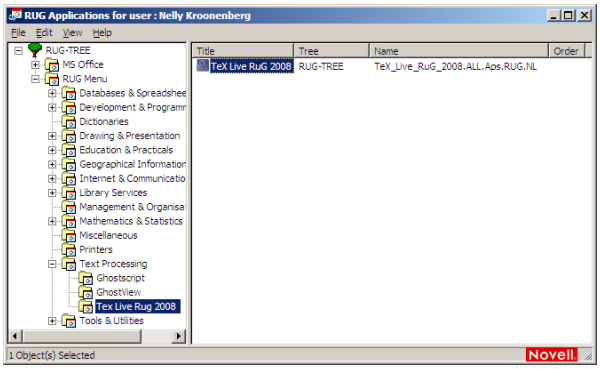


Figure 3. The NAL application menu

in configuration strategy in MiKTEX from version 2.4 to version 2.5. I understand that there are again major changes with MiKTEX 2.8.

CD edition. A companion CD was created. At first this contained a set of downloaded installers with directions for use. As a result, I was asked time and again to install TEX on people's laptops. Therefore, a later version got a modified version of the network installer. Nowadays, the CD is offered as an ISO image in the root of the installation.

Expanded user base. The user base for our MiKTEX was expanded first with students, then with other departments.

Standardization. Around the time of the second MiKTEX edition, we also moved to standardized desktops and roaming profiles; see the next section.

Graphics support. WMF2EPS was dropped, in part because it was shareware and I didn't want to deal with licensing issues for the expanded user base, in part for technical reasons. In its place, I created EPSPDF to take care of converting and cropping arbitrary PostScript (print)files.

The university network

For some time, we have had a centrally managed university-wide Novell network. Software and licenses are also centrally managed. There is a standardized Windows XP workstation. Standard software and additional software is installed from the network and where possible also run from the network. NAL (Novell Application Launcher) is the network component which takes care of this.

Figure 3 displays the NAL menu as seen from my computer at the university (the TEX Live entry merely points to the installation script).

Staff members can and do install software of their own on their local system if they want to. Students do not have this luxury. Some staff members download and install their own TEX.

The standard workstation is configured with *roaming profiles*, *i.e.* user configuration is mostly copied to the network on logout and copied back to the local system on login. Users see the same desktop on any client computer on the net, of course excepting software which staff members may have installed locally.

Roaming profile configuration should involve nothing local, unless it is copied to and from the network as part of the user profile. It should not require admin rights either. This is especially important for classroom computers and for students.

TEX Live

In 2008 I got involved in the TEX Live project. I mostly worked on Windows support, keeping an eye on the needs of our university installation.

I have done only a little work on the 2009 edition. However, other team members now also have Windows virtual machines for testing, and we have been joined by a real Windows user, Tomasz Trzeciak. He proved to us that DOS batchfiles aren't quite as lame as we thought they were.

Compared to MiKTEX 2.5, TEX Live is a lot simpler to turn into a network installation:[2] in good Unix tradition, TEX Live uses environment variables and plain text files for configuration.

Relocatable. An important function of configuration is telling programs where they can find the files they need. Normally, TEX Live puts only relative paths in the configuration files. Programs can combine these relative paths with their own location to determine absolute paths. With this strategy, configuration files can stay the same if the installation as a whole is transferred to another place.

Batteries included. TEX Live contains copies of Perl and Ghostscript for Windows. This puts Windows on a more equal footing with Unix/Linux with regard to all the scripted utilities that are part of a typical TEX installation.

Both the included Ghostscript and the included Perl are hidden, *i.e.* TEX Live knows that they are there, but the rest of the system doesn't. They are not on the search path, and there are no environment variables or registry settings created for them. Therefore, they shouldn't interfere with pre-installed copies. The only disadvantage is the disk space they take up. But this is hardly significant with today's hard disk sizes.

Creating the installation

I emulate the university networking setup by setting up a Samba server on my Linux machine. Its clients are virtual

machines.

Samba has been set up for roaming profiles. There is a share for the profiles, an X:-share for home directories and a Z:-share with applications, in exactly the same layout as the university.

I install TEX Live into the right position on the Z:-share by running the installer on my own Linux system. I select binaries for both Linux and Windows.

I switch between this and my regular installation simply by changing environment variables, for which I have a small shell function. This lets me do much testing and all maintenance from Linux. I explained already that configuration files don't depend on the location of the installation. So it doesn't matter that, seen from Linux, the installation is in a totally different place than it is as seen from Windows.

I populate the texmf-local directory tree with the university house style and some legacy packages. It was almost a straight copy of the corresponding local tree from the previous MiKTEX-based installation. For students, there is no need for a local tree.

The 2009 installer

The network installer doesn't have to do much: there are hardly any options, it doesn't have to download and install packages, it just has to add TEX Live to the search path, create some shortcuts, register an uninstaller and optionally create some file associations.

For most of this, it can use the library functions of the installer and the TEX Live Manager, both of which are written in Perl.

The following code adds TEX Live to the search path and creates some menu shortcuts:

```
#!/usr/bin/env perl
BEGIN {
  require "tlmgr.pl";
}

# Only make user-level changes even if admin
$opts{'w32mode'} = 'user';

# Note. The action_... functions read
# their arguments from @ARGV.

# Add TeX Live to path
unshift @ARGV, 'add';
action_path();

# create some shortcuts
unshift @ARGV, 'install', 'shortcut',
 'dviout.win32', 'texworks', 'texlive-en',
 'tlpsv.win32';
action_postaction();
```

File associations can be done similarly. A corresponding uninstaller script:

```
BEGIN {
  require "tlmgr.pl";
}
$opts{'w32mode'} = 'user';

# remove shortcuts
unshift @ARGV, 'remove', 'shortcut',
 'dviout.win32', 'texworks', 'texlive-en',
 'tlpsv.win32';
action_postaction();

# Remove TeX Live from path
unshift @ARGV, 'remove';
action_path();
```

Registering and unregistering the uninstaller. However, it is a bit more complicated to register one's custom uninstaller. The TEX Live modules in `tlpkg/TeXLive`, and the modules they load, contain everything needed, but the interface is comparatively low-level. Here is the code, for what it is worth:

```
# don't need to re-require modules but
# do need to re-import names
Win32::TieRegistry->import(qw($Registry));
$Registry->Delimiter('/');
$Registry->ArrayValues(0);
$Registry->FixSzNulls(1);

# register uninstaller. Failure not fatal.
my $Master_bsl = $Master;
$Master_bsl =~ s,/,\\,g;

my $rootkey = $Registry -> Open("CUser",
 {Access =>
   Win32::TieRegistry::KEY_ALL_ACCESS()});
my $k;
if ($rootkey) {
 $k = $rootkey->CreateKey(
  "software/microsoft/windows/" .
  "currentversion/uninstall/OurTeXLive/");
 if ($k) {
  $k->{"/DisplayName"} = "OurTeXLive 2009";
  $k->{"/UninstallString"} =
    "\"$Master_bsl\\w32unclient.bat\"";
  $k->{'/DisplayVersion'} = "2009";
  $k->{'/URLInfoAbout'} =
    "http://ourwebsite.edu/ourtexlive";
 }
}
```

```
warn "Failed to register uninstaller\n"
 unless $k;
```

and for unregistering the uninstaller:

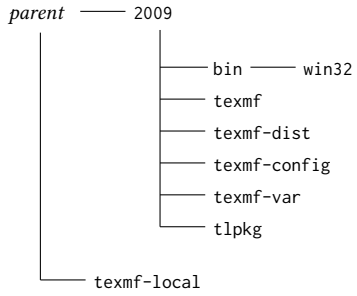
```
my $rootkey = $Registry -> Open("CUser",
 {Access =>
   Win32::TieRegistry::KEY_ALL_ACCESS()});
if ($rootkey) { # otherwise fail silently
 my $k = $rootkey->Open(
  "software/microsoft/windows/" .
  "currentversion/uninstall/");
 TeXLive::TLWinGoo::reg_delete_recurse($k,
  'OurTexLive/') if $k;
}
```

Prototype installer scripts are available at `http://tug.org/texlive/w32client.html`.

ZENWorks. Novell has a tool ZENworks for repackaging applications for NAL. It tracks changes in the registry and the filesystem during installation. The ZEN-generated installer of the repackaged application duplicates those changes. However, for me it was more practical to use a Perl script, and I am grateful to the ICT people that they let me.

Directory layout

We assume the standard TEX Live directory layout, with texmf-local one level higher than the other trees:



It is possible to choose another layout, *e.g.* without the year level, but then the components of TEX Live need a bit more help to find their support files.

Batch wrappers

We also need a wrapper batchfile to make sure that the Perl from TEX Live is used rather than a locally installed Perl, and to take care that `tlmgr.pl`, the TEX Live Manager Perl script, is found. This file is used as a library by our custom installer. Below is a bare-bones wrapper batchfile; in the standard TEX Live we use much more involved wrappers, with various safeguards.[3]

```
set this=%~dp0
rem Use TL Perl
```

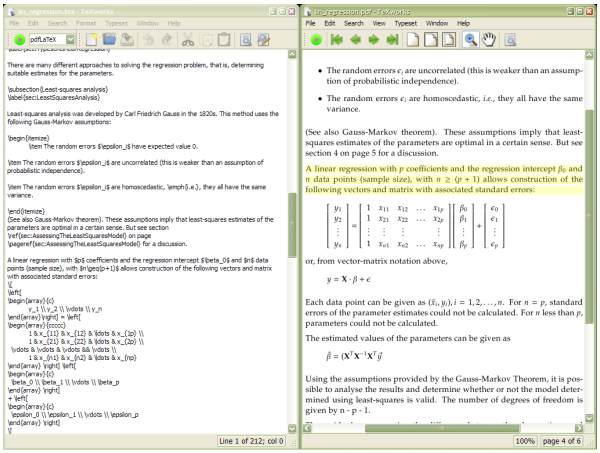


Figure 4. TeXworks for Windows is included in TEX Live

```
path %this%tlpkg\tlperl\bin;%this%bin\win32;
    %path%
rem (one line)
set PERL5LIB=%this%tlpkg\tlperl\lib;
    %this%tlpkg;%this%texmf\scripts\texlive
rem (one line)

rem Start Perl script of the same name
perl "%~dpn0" %*
rem Give user opportunity to scan output msgs
pause
```

Note the first line, where the batchfile finds its own directory: `set this=%~dp0`. This syntax has been available since Windows NT.

The `w32client` and `w32unclient` scripts assume that they are in the root of the installation, *i.e.* in *<parent>*`/2009`. However, this is easy to change.

Getting TEX Live on the network

The first step was asking the ICT department for a directory to install into, preferably with write access for me, so that I can do maintenance without having to involve ICT every time.

The next step was copying everything to that directory. For transport, I nowadays make a giant zip of the installation and put it on a USB stick.

Finally, the installer is integrated into the Novell NAL menu; see figure 3 on page 87. This is done by ICT staff.

The installer places the TEX Live menu itself under Start / Programs, just as the native installer does.

For maintenance, I used to do small changes manually and do larger changes by wholesale replacement. For the future, rsync looks like a better way.

Additional software

TEX Live by itself is pretty complete. For Windows, it includes the TEXworks cross-platform editor (figure 4), and the PS_View PostScript viewer, which can also read pdf. As mentioned earlier, it also contains private copies of Unix mainstays such as Perl and Ghostscript that many TEX Live components depend upon.

Nevertheless, some additions would be desirable: another editor besides TEXworks, more graphics support, maybe a bibliography editor.

But there are requirements for such add-ons:

- Free (as in beer)
- Per-user configuration
- Usable for non-geeks

Several programs looked interesting, but didn't meet these requirements or had other problems. They include alternative LATEX editors TEXmaker and WinShell, bibliography editors JabRef (Java-based) and BibEdt, and a couple of draw programs, IPE and TpX. So I followed the example of previous TEX Live DVDs and put installers for them in a `support` subdirectory. Frankly, I don't know whether anybody has made use of these installers.

Editor. For 2008, I decided to stick with TEXnicCenter as editor. I wrote some fairly elaborate code to configure it for TEX Live, since the automatic configuration didn't work as nicely for TEX Live as it did for MiKTEX. I also looked at TEXmaker. It would have been much easier to configure, but at that time it still lacked some important features.

For the 2009 release I'll keep TEXnicCenter, if only because many users dislike change, but I'll also include TEXworks, which is already part of standard TEX Live.

Documentation. The TEX Live menu contains various shortcuts to manuals, such as the UK FAQ and the 'not so short introduction'. There are also links to the CTAN catalogue and to my own web page for this installation, `http://tex.aanhet.net/miktex/` (!).

Vista and Windows 7

Strictly speaking, this topic doesn't belong here: the network installation only targets XP machines. However, for the standard TEX Live we had to make sure it would work properly with Vista and Windows 7. Testing this, we ran into some interesting problems.

UAC. Vista introduced User Account Control, or UAC in short. This means, among other things, that only administrators are allowed to install software under Program Files, and only administrators are allowed to change the more important system settings in the registry.

A new slightly annoying twist is that even administrators don't have these privileges automatically. To start a program with admin privileges, you can right-click the shortcut, which usually gives you an option 'Run as administrator'. An administrator has to confirm his intentions, a non-administrator has to provide administrator credentials at this point.

Virtualization. But there is another, more insidious twist: Vista/Win7 may guess that a program needs administrative privileges, *e.g.* because it has 'install' or 'setup' in its name. If such a program wasn't started with administrative privileges, Vista may fake them. In particular, attempts to write to Program Files might result in writings to *user*`\appdata\local\virtualstore`. For registry access, similar virtualization might be applied.[4]

Installing TEX Live with real admin privileges and adding packages with faked admin privileges is not healthy for a TEX Live installation.

This compatibility mode can be avoided by the addition of a *manifest* which is a bit of XML that explicitly tells Windows under which privileges the program needs to be run. The options are (explanations literally taken from `msdn.microsoft.com`):

asInvoker The application runs with the same access token as the parent process.

highestAvailable The application runs with the highest privileges the current user can obtain.

requireAdministrator The application runs only for administrators and requires that the application be launched with the full access token of an administrator.

This XML can be embedded into the binary or added as a separate file with the same name as the program, but with `.manifest` appended.

This is our manifest file for the Windows Perl executable:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1"
    manifestVersion="1.0">
  <assemblyIdentity
    version="1.0.0.0"
    processorArchitecture="*"
    name="perl.exe"
    type="win32"/>
  <trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
    <security>
      <requestedPrivileges>
        <requestedExecutionLevel level="asInvoker"/>
      </requestedPrivileges>
    </security>
  </trustInfo>
</assembly>
```

We believe that, with the addition of a couple of manifest

files and some tests on admin privileges, TEX Live 2009 has become Vista-safe.

Conclusion

So you see that maintaining a TEX installation has little to do with TEX, slightly more with programming, but is mostly a matter of tying disparate pieces together.

Notes

1. MAPS 33 pp. 59–64 and TUGboat 27:1 pp. 22–27.
2. MiKTEX 2.8 may be easier to deal with, but I didn't check this out.
3. TEX Live even uses a second, binary wrapper around the batch wrapper because some programs handle batchfiles badly.
4. This only happens on 32-bit Vista/Win7.

Siep Kroonenberg
n.s.kroonenberg@rug.nl