# Notes on: Tuque for OS/X and Linux

Erick Engelke, September 20, 2017  
Notes Version 1.0.0

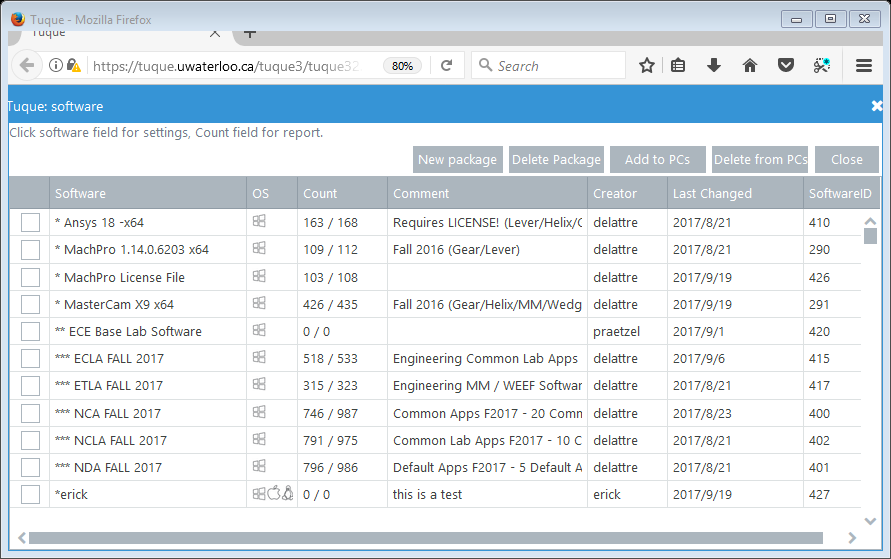
This document assumes you have a working knowledge of Tuque when used for Windows computers and are now thinking of using it to manage OS/X 32/64 and Linux 64 bit workstations.

Some modern features of Tuque are:

* GUI Compatible with all web browsers, including tablets, and in a pinch: phones
* GUI Easy to learn
* Groups of packages can be added to groups of workstations, so you can add all required software packages into EngCompFall2017, and deploy them to EngCompAllLabs with a single click
* Packages can be members of multiple super/group packages
* Workstations can be members of multiple groups eg. EC Lever Lab, EngCompAllLabs, etc.
* Adding new workstations to an existing group picks up the software meant for that group
* Special groups can be created for situations like optical scanners in multiple other rooms, only scanner-denoted workstations will receive the scanner software
* Detailed reporting in real time

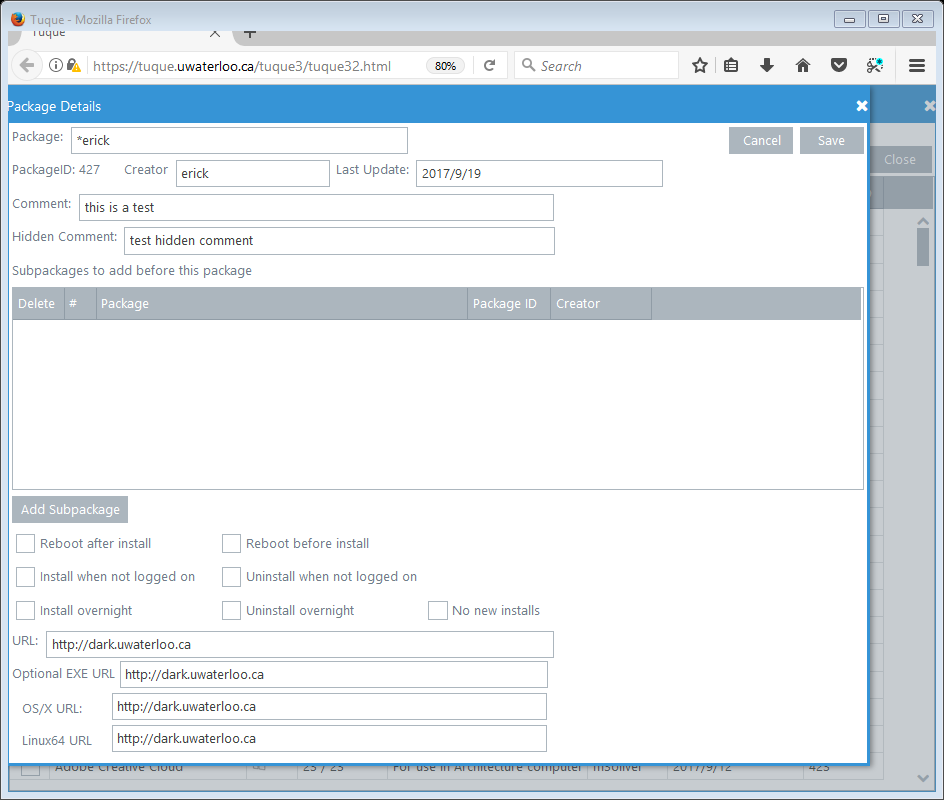
As of version 3.2, Tuque supports multiple client operating systems. So if a group (eg. EngDean) all get Chrome, MS-Office, Acrobat Reader, etc., OS/X clients of that same group will receive OS/X versions of the same applications distributed to Windows clients.

When you list software in Tuque, some packages will display a Windows icon or OS/X or Linux or some combination of the three.



For whichever OS entries are displayed, there is a tuque application bundle specific for that technology. So it makes sense that there are three separate bundles for the package *\*erick* above, which claims to support all three OSes.

Clicking on the *\*erick* package, we see that there are indeed separate URLs specified for each operating system. In this case they all three point to the same URL, though in practice they could and will usually point to three separate URLs.



Almost everything you know about Tuque for Windows applies to the Posix systems (OS/X and Linux), in fact they share the same codebase, so they should mostly stay in sync as we move forward.

The major differences are as follows:

**Bitness**

The OS/X binary is presently 32 bit but can install 32 or 64 bit packages, we will have the option of a 64 bit executable in a few months, for now 32 bits works everywhere. Linux is only 64 bit – but that’s compatible with all modern distributions, and it can distribute 32 and 64 bit packages.

**Directories**:

**Windows**

C:\nexus\install tuque uses for a variety of things

C:\nexus\install\work the working subdirectory that gets erased later

C:\nexus\install\log log file

C:\nexus\install\work\tuqueresults.dat format: integer:message

**Posix**

/var/tuque

/var/tuque/work

/var/tuque/tuque.log

/var/tuque/work/tuqueresults.dat format integer:message

**Script**

**Windows**

C:\nexus\install\work\install.cmd

C:\nexus\install\work\uninstall.cmd

**Posix** The filenames were chosen so posix ZIP files could be multiplatform.

/var/tuque/work/osxinstall.rc

/var/tuque/work/osxuninstall.rc

/var/tuque/work/linuxinstall.rc

/var/tuque/work/linuxuninstall.rc

**Program**

Windows normally uses a service (**tuqueinstaller1.exe**, net start tuqueinstaller). There is also a one-time program called G**oTuque.exe** which can be run *by Administrator instead of the service*.

GoTuque.exe is suitable for laptops which need to be populated with software but will not poll for updates after the install.

Posix systems use a slightly modified console application GoTuque which must be run by *root*, or *sudo* or similar. It can be run at startup, or by ssh or a cron job or manually.

For the first few months, OS/X systems will have to run MacGetTuque which is a GUI application fitted over the GoTuque application. A vendor we rely on will not be compatible with console OS/X executables until later 2017 at which time we will likely convert to the console GoTuque strategy for OS/X.

**Zipped Packages**

For all platforms, tuque stores the data in a large zip-64 bit file. Posix users might have expected tar.Z files, but for compatibility with the whole system the decision was made to stick to one technology: ZIP. Also, ZIP is natively available and compatible across all three platforms.

**Scripts**

The osxinstall / osxuninstall / linuxinstall / linuxuninstall files are all expected to be scripts with their first lines having the Posix magic #! Characters. In other words, normal Posix scripts !

The scripts are expected to not return until the installation is complete (or completely removed for uninstalls).

Posix Tuque supplies /var/tuque/work/*manyfiles including shell scripts.* It is the script writer’s option to use any standard out-of-the-box-compatible technology on those files. Note, all files will be mode 0644 on the system.

Tuque automatically converts the installing script to mode 0700 and executes it. There should be no assumptions of which directory is active, so you should change to the preferred work directory: /var/tuque/work

**OS/X Scripts**

We recommend using one of the popular OS/X installation tools like Composer to bundle the package into a couple of files. If your tools create a DMG image, that’s okay, we will still zip it up even though the compression may already be present – ZIP technology is used to bundle the files into one downloadable package file.

**Linux Scripts**

Linux scripts present a challenge because they should be compatible with all popular distros. In particular: Centos/Redhat and Ubuntu.

For example, a Linux script might look for Centos/Redhat and use appropriate package tools from those distros, and also detect and implement the appropriate package calls for Ubuntu.

You have the option of including the necessary files or letting the package utility download them. However, it would be optimized for speed and network usage if you placed the necessary files into the ZIP package.

**Learning Scripting**

It is possible to look at other people’s Tuque packages and read what they have done. We will try to collect and share best practices as they are developed.

Eg.

#!/bin/bash

# tuque may pass us a work directory, if not assume it is here

if ["$WORKDIR" = ""]; then

WORKDIR="/var/tuque/work"

fi

# setup ourselves

cd $WORKDIR

/bin/pwd

# install a file

/bin/pwd > $WORKDIR/test.dat

# defaults to success but you can specify an override

# set the error results

echo "101:no permissions" > tuqueresults.dat