Subject: WebUpdater
Posted by Mindtraveller on Wed, 19 May 2010 09:24:59 GMT
View Forum Message <> Reply to Message

I think it is time for U++ to become more mature and more "industrial standard" as a tool for making not only effective, but full featured modern level apps.

Initially U++ is bundled with Updater package, which is discussed in one of parallel topics. Updater is intended to run only on LAN in controlled environment. This is good, but today's applications undoubtedly require auto web-update function.

OK, so the idea is to make WebUpdater package to make app updatable from parent web resource.

So, my ideas:
1. WebUpdater should work through HttpClient (with standard http request).
FTP fits less here as it requires adding anonymous account which is not good generally. Using HTTP makes it possible to download in a way everybody does, so, i.e. these downloads may be counted and added to stats with engines like Drupal. Also HttpClient supports proxy settings which is very good too.

2. Parent web resource should contain rather small file with sizes and MD5 hashes of all updatable files. WebUpdater checks current MD5 of local copies, then download MD5-file and compares. And all this in background. This file may also contain short description of new version to be displayed in the host app.

3. WebUpdater should support partial downloading to minimize potential traffic for users. Of course, this could be done with comparing current temporary downloaded file sizes with ones from downloaded file.

4. WebUpdater should contain global object which optionally executes WebUpdater.exe after host program is closed. WebUpdater.exe is simple tool which checks downloaded files' MD5. If everyone is OK, just moves them into working dir.

5. Under POSIX systems, if app was installed from system software manager, WebUpdater just downloads new version package and executes system manager for it. It is much better here, because this will solve many issues like required root access or packages management.

What do you thing about it?

Subject: Re: WebUpdater
Posted by dolik.rce on Wed, 19 May 2010 09:57:30 GMT
View Forum Message <> Reply to Message

Definitely important and good idea.

For me, especially no.5 is interesting. It won't be easy, because there is many systems with many different packaging strategies and even more managers. It will take quite some work to cover at least the main POSIX systems. On unsupported systems we will have to fallback to copying files. Also, some managers (e.g. Arch Linux pacman) require to be started with root permissions, so it doesn't really solve the access problem.

Anyway, count with my help with the packaging systems I'm familiar with

Honza

Subject: Re: WebUpdater
Posted by mr_ped on Wed, 19 May 2010 10:27:00 GMT
View Forum Message <> Reply to Message

Well, in ideal world some of the things has to pushed a bit farter (although your list is quite exhaustive already):

1) make the downloader modular (HttpClient behind abstract interface), so it can be exchanged with rsync/zsync/bittorrent/... modules later (designing such unified API will be tricky though). Also some modules may support user authentication, so you can make sure your update server does service only registered users.

2) there's no security incorporated, I think this is under "industrial standard" (or better to say the industrial standard is too low, as I'm looking around at some industry solutions around me).
There are several levels where checking is desirable:
- download integrity (md5 sums as you proposed)
- user is authorized to upgrade to desired version (A)
- parent server verification (B)
- encrypted communication with server (C)

A) things to consider
- local user's installation vs global one
- corporate permissions, like allowing/forcing certain versions per user, etc. (implementation shouldn't be part of this updater, but there should be interface/hooks for this?)

B) you have to verify the update server is the real one. I think the digital signing of final update files is a must, eventually signed revision files with versions+md5sums would be good to discover problems before file downloading.

Also md5sum as is is already too weak, I would used at least sha1+md5 or sha2 class (+md5 doesn't hurt).

C) the https or other encrypted downloader would be nice, although that can be worked around by using 7z with password and sending only zip files between app and server, so there are no public data for somebody eavesdropping. (but https would be more robust, as the encrypted zip files are useless once the password leaks .. still better than nothing) hmm.. this should help to cover both open applications (where only the integrity is a concern) and closed applications where you want to have server on internet, but you don't want to give away any info to guests, only to your registered customers (plus you still need the integrity checks anyway).

Subject: Re: WebUpdater
Posted by mr_ped on Wed, 19 May 2010 10:31:01 GMT
View Forum Message <> Reply to Message

Well, using .debs does solve many things with integrity, but then again if you are already building .deb packages for your app, it's best to set up your repository and let the OS fetch updates trough normal update procedure. I think using custom updater will be counter-productive in such case, it would be better to add just some repository setup wizard and maybe using it to check user's settings periodically. But let the update alone to be done by ordinary means.

Subject: Re: WebUpdater
Posted by koldo on Wed, 19 May 2010 10:50:25 GMT
View Forum Message <> Reply to Message

It is a good idea.

Subject: Re: WebUpdater
Posted by fudadmin on Wed, 19 May 2010 11:23:50 GMT
View Forum Message <> Reply to Message

Off topic. Misspelling of the week:
Quote:
the things has to pushed a bit farter (although ... is quite exhaustive already):
