Improved Linux Download Scripts

by Hartmut Buhrmester

Introduction

I like to introduce a complete rewrite of the Linux download scripts for the project WSUS Offline Update. These scripts offer many improvements over the legacy script DownloadUpdates.sh:

- Separation of a frontend and backend script

  The script update-generator.bash is used to interactively select the update, language and download options. The script download-updates.bash fetches the selected updates without any user interaction. This separation makes the structure of both files more straightforward.

- Highly modular approach

  Both scripts are further split into libraries, common tasks, setup tasks and download tasks. Each script does one task only in the most straightforward manner. This resembles the flow of control and makes the scripts easily expandable and more maintainable.

- Unified language settings

  There is no distinction between default languages, custom languages and update languages.

  Users can specify *one* language on the command line, and then they will get downloads for the specified language only, and nothing more.

  Note: Recent versions of the Linux download scripts allow to specify multiple languages on the command-line as a comma-separated list. The general approach stays the same: Only the specified languages are downloaded—nothing more or less.

- Verification of downloaded files

  SHA-1 hashes are embedded into the filename of all security updates, as a number of 40 hexadecimal digits. These are compared to the checksums, which are calculated by hashdeep.

  The verification of digital file signatures with Sysinternals Sigcheck running under wine was tried, but it doesn't really work without the necessary root certificates.

- Compatibility

  The download script uses the same algorithms for calculating superseded and dynamic updates as the Windows script DownloadUpdates.cmd. The compliance with the Windows scripts can be tested with the scripts compare-integrity-database.bash and compare-update-tables.bash.

- Desktop integration

Obsolete updates are not deleted immediately, but moved into the trash. GNOME and most other GTK+ based desktop environments use GVFS to handle the trash. The package trash-cli can be used with other desktop environments or window managers. trash-cli should also work without any graphical environment.

- Self updates of WSUS Offline Update

  Both the setup and the download script check for new versions of WSUS Offline Update. They also handle updates of the configuration files in the static and exclude directories.

- Same day rules

  Same day rules are used to prevent the repeated evaluation of the same tasks in adjacent runs of the download script.

- Documentation

  There is even a complete documentation.

Compatibility

The scripts are only tested on:

- Debian 6.0 Squeeze
- Debian 7 Wheezy
- Debian 8 Jessie
- Debian 9 Stretch
- Debian 10 Buster, the current *stable* since July 2019

Other Linux distributions should just work fine, if the needed applications are installed (see below).

FreeBSD should work, because it seems to use many GNU utilities. For example, the manual page for *grep* refers to the GNU Project. The command readlink in FreeBSD seems to be compatible with that in Linux.

- https://www.freebsd.org/cgi/man.cgi

There are still many small differences between BSD and GNU core utilities like *date* and *sed*. These were reviewed in version 1.16 from 2020-01-08.

Mac OS X (macOS) is different: it uses some GNU utilities, but they all stay at GPL v2. Everything with GPL v3 is shunned by Apple. This means, that the bash will stay at version 3.2.53 forever. The bash was eventually replaced with the zsh, which uses a more permissive license.

The download script download-updates.bash may still work, though, because it doesn't use more recent features. Also, developing for a plain POSIX shell is not the answer, since this affects all other commands as well. For example, the command readlink in Mac OS X is *not* compatible with the same command in Linux, and there are long discussions in the Internet about this one utility:

- https://stackoverflow.com/questions/1055671/how-can-i-get-the-behavior-of-gnus-readlink-f-on-a-mac

Requirements

The download script uses some additional applications, which can usually be installed from the repositories of the Linux distribution.

Required applications

- **cabextract** is used to extract the file package.xml from the WSUS catalog file.
- **hashdeep** is used to create an integrity database of the downloaded files. This allows a simple integrity check: For most security updates, the SHA-1 hash is embedded into the filename as a hexadecimal number of 40 digits. For example, if the filename is `ndp35sp1-kb958484-x64_e69006433c1006c53da651914dc8162bbdd80d41.exe`, then the SHA-1 hash of the file is `e69006433c1006c53da651914dc8162bbdd80d41`. After calculating the hashes with hashdeep, they can be easily compared to the expected values.

  The application hashdeep is provided by the package md5deep or hashdeep, depending on your distribution. See the *Installation* section later in this document.
- **unzip** is used for the self-update of the WSUS Offline Update installation and to unpack the Sysinternals utilities Autologon and Sigcheck.
- **wget** is the standard interactive download utility for Linux. It should be installed by default in Linux distributions, but this may not be true for Mac OS X and other BSDs.
- **XMLStarlet** is used to extract information from the file package.xml, to calculate dynamic and superseded updates. It is provided by the package xmlstarlet.

  Note, that the installed binary may be `/usr/bin/xmlstarlet` in Debian and Red Hat, or `/usr/bin/xml` in distributions, which use the unmodified upstream source from http://xmlstar.sourceforge.net/ directly. Despite such possible differences, the package name should always be *xmlstarlet*, and there never was a package *xml*.

Recommended applications

- **dialog** is used by the script update-generator.bash to create nicely formatted selection dialogs in the terminal. It allows multiple selections for the Windows or Office versions, languages and optional downloads. Then all needed updates can be downloaded with just one run of the backend script download-updates.bash.

  If the external command *dialog* is not installed, then the selection dialogs will be created with the internal command *select* of the bash. This only allows single selections.

- **rsync** is used by the script copy-to-target.bash, to synchronize the client directory with another directory, for example on a USB drive.
- **gio trash, gvfs-trash** and **trash-put** can be used to move old files into the trash, rather than deleting them directly.

  The virtual file system gvfs is used by GNOME and other GTK+ based desktop environments. gvfs-trash is provided by the package gvfs-bin in Debian 9 Stretch/stable. In the upcoming Debian 10 Buster/testing, gvfs-trash is *deprecated* and gio from the package libglib2.0-bin should be used instead.

  trash-put is provided by the package trash-cli. These Python scripts implement the FreeDesktop.org Trash specification. They can be used with other desktop environments and window managers, and they should even work without any graphical user interface. If the package trash-cli is not available from the package repository of the Linux distribution, it may be installed from the upstream Python repository with:

  ```
  pip install trash-cli
  ```

  The local trash directory is `$HOME/.local/share/Trash/`.

  Creating a trash directory on external drives requires sufficient rights. It will be an invisible directory `.Trash-1000` at the root level of the drive. The number 1000 is the user ID of the first regular user on Debian. It will be `.Trash-500` on Fedora.

Optional applications

- **aria2** features multiple connections, to speed up the download of large files. The time stamping feature of aria2 usually works better than that of wget 1.16 and previous versions.

  wget 1.16 always used *two* queries for each download: a HEAD query to get the file size and modification date of the remote file, and a GET query to download the file, if the server file is newer than the local file. wget 1.16 also downloads the file again, whenever the file size changes, regardless of the file modification date. In a content delivery network, and with files, which change very often like the virus definition files, this may lead to all kinds of strange behavior, because wget may get different answers from different servers: Sometimes wget downloads an older file, or it may just download the same file again.

  aria2 only uses *one* GET query, along with a conditional header If-Modified-Since. Then the server must decide, if the server file is newer than the local file. The server should respond with either “304 Not Modified” or “200 OK”. If the answer is “200 OK”, then the newer file will be downloaded.

  But there is a **bug**, which was observed a few times: Sometimes the Microsoft servers ignore the conditional header and *always* return “200 OK”. Then aria2 proceeds to download the same file again.

wget 1.18 in Debian 9 Stretch uses the same approach as aria2 for time stamping: It also uses a single GET request with a conditional header If-Modified-Since. But it also checks the answer and compares the file modification date itself. If some Microsoft servers choose to ignore the conditional header, then wget recognizes the wrong answer “200 OK” and does not download the same file again. In this case, you may find these messages in the download log:

```
Connecting to download.windowsupdate.com …
HTTP request sent, awaiting response... 200 OK
Server ignored If-Modified-Since header for file '…'
You might want to add --no-if-modified-since option.
```

Therefore, **wget 1.18** or higher is the recommended download utility. The mentioned option `–no-if-modified-since` is not used, though, because this would restore the behavior of wget 1.16, and this only causes other problems.

Note, that the application aria2 is provided by the package aria2 in Debian, but the installed binary is `/usr/bin/aria2c`.

If you like to use aria2, change the search order in the file preferences.bash to:

```
supported_downloaders="aria2c wget"
```

- **wine** can be used to run Sysinternals Sigcheck on Linux. This could be used to validate digital file signatures, but it doesn’t work so far. See a discussion below in the chapter *Validation of downloaded files*.
- **mkisofs** or the fork **genisoimage** is needed for the script create-iso-image.bash.

Downloads

The current version is 2.0, which was released on 2020-03-25. It is compatible with WSUS Offline Update 12.0.

Please see the introduction of the Linux download scripts in the WSUS Offline Update Forum for possible updates:

- Introducing a complete rewrite of the Linux download scripts

Installation

WSUS Offline Update already includes the new Linux download scripts. You don't need to install the Linux scripts separately, as it was necessary for the first beta versions, but you should review the needed packages from your Linux distribution.

Install the required and recommended packages

For Debian and Debian-derived distributions, you need to distinguish between the packages *md5deep* and *hashdeep*.

The upstream developers moved their project from SourceForge to GitHub, and they renamed their project from md5deep to hashdeep:

- http://md5deep.sourceforge.net/
- https://github.com/jessek/hashdeep/

Debian followed this move and renamed the package md5deep to hashdeep, starting with Debian 8 Jessie-Backports in summer 2015. The general rule for Debian and Debian-derived distributions then is: Install the package md5deep, if the distribution was released before 2015. Install the package hashdeep for all recent distributions.

- For Debian 7 Wheezy:

```
su -
aptitude install cabextract md5deep unzip wget xmlstarlet \
                 dialog genisoimage rsync trash-cli
```

- For Debian 8 Jessie-Backports and newer:

```
su -
aptitude install cabextract hashdeep unzip wget xmlstarlet \
                 dialog genisoimage rsync trash-cli
```

- For Ubuntu 14.04 LTS Trusty:

```
sudo apt-get install cabextract md5deep unzip wget xmlstarlet \
                     dialog genisoimage rsync trash-cli
```

- For Ubuntu 16.04 LTS Xenial and newer:

```
sudo apt-get install cabextract hashdeep unzip wget xmlstarlet \
                     dialog genisoimage rsync trash-cli
```

Other distributions, which are not Debian-based, seem to keep the package name md5deep.

- For Fedora 27 (thanks to *username*):

```
sudo dnf install -y cabextract md5deep unzip wget xmlstarlet \
                    dialog genisoimage rsync trash-cli
```

- For FreeBSD 12.1 (thanks to *TheFlipside*):

```
su -
pkg install bash cabextract md5deep wget xmlstarlet \
            cdrkit-genisoimage rsync
```

Note, that both packages md5deep and hashdeep install a series of related applications: hashdeep, md5deep, sha1deep, sha256deep, tigerdeep, and whirlpooldeep. Throughout WSUS Offline Update, you always need the application *hashdeep*, regardless of the package name.

Install optional packages

The packages listed above are necessary to run the Linux download scripts in their default configuration. There are some optional features, which require the installation of additional packages:

- The verification of digital file signatures only works halfway, because the Microsoft root certificates are not available in Linux. If you still like to try, you need to install **wine** to run Sysinternals Sigcheck.

  Note, that most security updates can also be verified by comparing the SHA-1 hashes, which are plainly inserted into the filenames, with the values, which are calculated by hashdeep.

- The download utility aria2 features multiple simultaneous downloads, which may be useful for slow connections. It requires the package **aria2**. But see the discussion above for a known bug.

Download and unpack the wsusoffline archive

Download the newest wsusoffline archive from the download page https://download.wsusoffline.net/ and unpack it. Note, that the zip archive comes with an accompanying hashes files. You can use it to verify the download with:

```
hashdeep -a -v -v -l -k wsusoffline1161_hashes.txt wsusoffline1161.zip
```

The new Linux scripts are included in the "sh" subdirectory. Due to the packaging on Windows, the scripts are not yet executable. Run the included script fix-file-permissions.bash once as:

```
bash fix-file-permissions.bash
```

to make the scripts update-generator.bash, download-updates.bash, get-all-updates.bash, and some others executable.

You can then use the script update-generator.bash to interactively select your updates, languages and optional downloads.

You can use the script get-all-updates.bash as a template: This script downloads all updates with all available options for the default languages German and English. But it is also meant for customization – you can simply comment out or delete all items you don't need.

Once the scripts are executable, you can run them from the script directory with:

```
./update-generator.bash
./get-all-updates.bash
./download-updates.bash w60 deu,enu -includesp
```

Notes

The new Linux scripts don't work alone – they need the configuration files from the wsusoffline installation. Also, the Linux download scripts can only replace the Windows download scripts, e.g.

DownloadUpdates.cmd. To install the updates, you surely need the files in the client directory, e.g. the application UpdateInstaller.exe.

Therefore, you should not download the Linux scripts separately, as it was necessary for the first beta versions. Just get the latest wsusoffline archive and find the Linux scripts in the "sh" subdirectory.

If you need to copy or move the wsusoffline directory, please make sure to keep the modification date of all files. You could use "`cp --archive`" or "`cp --preserve`" instead of just "`cp`". This is important for all files throughout WSUS Offline Update.

Configuration

The download scripts don't need an initial configuration. You can, however, edit some permanent settings in the optional preferences file.

This preferences files is provided as a template preferences-template.bash. It must be renamed or copied to preferences.bash to be used. This is meant as a simple way to prevent customized settings from being overwritten on each update.

Since the preferences file is optional, other scripts like download-updates.bash and update-generator.bash provide the default settings for the preferences file – but they should not be edited there. The intended usage is to create the file preferences.bash from the template and make changes there.

The preferences file is extensively commented. You can change:

- The preferred download utility: GNU wget or aria2
- Proxy servers can be set in the file preferences.bash, but there are more ways to do so: Desktop environments like GNOME and KDE may use their own settings for proxy servers. You can also set the environment variables http_proxy and https_proxy in the file `~/.profile` or edit the preferences files for the download utilities wget and aria2.
- If you don't need Silverlight, you can set the option `include_win_glb` to *disabled*.

Monthly quality versus security-only updates

By default, WSUS Offline Update downloads and installs the full monthly *quality* update rollups for Windows 7 and Windows Server 2008 R2, Windows Server 2012, Windows 8.1 and Windows Server 2012 R2. To prefer *security-only* update rollups, change the option `prefer_seconly` to *enabled* in the file preferences.bash.

The switch from *quality* to the *security-only* updates uses two sets of configuration files. The files

```
../client/exclude/HideList-seconly.txt
../client/exclude/custom/HideList-seconly.txt
```

exclude the monthly *quality* updates from download, while the files

```
../client/static/StaticUpdateIds-w61-seconly.txt
```

```
../client/static/StaticUpdateIds-w62-seconly.txt
../client/static/StaticUpdateIds-w63-seconly.txt
../client/static/custom/StaticUpdateIds-w61-seconly.txt
../client/static/custom/StaticUpdateIds-w62-seconly.txt
../client/static/custom/StaticUpdateIds-w63-seconly.txt
```

include the *security-only* update rollups in download and installation. The files in the directories `../client/exclude` and `../client/static` are maintained by the author of WSUS Offline Update. They can be replaced at any time. Their counterparts in the directories `../client/exclude/custom` and `../client/static/custom` can be created for manual customization.

Overviews for the *quality* and *security-only* update rollups can be found at Microsoft:

- Windows 7 SP1 and Windows Server 2008 R2 SP1 update history
- Windows Server 2012 update history
- Windows 8.1 and Windows Server 2012 R2 update history

These Windows update rollups include updates for the .NET Frameworks, which are preinstalled in the respective Windows versions.

If other versions of the .NET Frameworks are installed manually, they must be updated separately. Since October 2016, these .NET updates are also released as full *quality* or *security-only* updates. They are announced the the .NET Blog:

- .NET Blog
- .NET Framework Monthly Rollups Explained

Viewing progress with wget

With GNU Wget 1.16 and lower, all messages are written to the log file. There is no progress indicator in the terminal window, in which the script is run. It is recommended to open another terminal window and view the progress with:

```
tail -F ../log/download.log
```

With GNU Wget 1.18 and later, a progress bar is shown in the terminal window, while the rest of the output is written to the log file. You may still watch the log file for details, e.g. for any download problems.

Usage

New users should just run the script `update-generator.bash` to interactively set up the download. This script doesn't have any command-line options. Just run it as:

```
./update-generator.bash
```

After selecting the updates, languages and optional downloads, the setup script shows the download command for confirmation, and then passes execution to the download script.

The script `download-updates.bash` is meant to run without any user interaction. It will ask for confirmation, if there are new versions of WSUS Offline Update or the Linux download scripts available, but this questions defaults to *no* after 30 seconds. This answer can be changed to *yes* by setting the option `unattended_updates` to *enabled* in the preferences file.

Once you are familiar with the different settings, you could customize the script `get-all-updates.bash` to get the downloads you need.

Language settings in the Windows scripts

The language settings for the Windows script DownloadUpdates.cmd are quite complicated, as it distinguishes between *default* languages, *custom* languages, and *update* languages.

All Windows versions since Windows Vista are considered to be *global*, but they still include localized installation files for:

- Internet Explorer 9 on Windows Vista
- Internet Explorer 11 on Windows 7
- .NET Framework language packs for languages other than English
- Microsoft Security Essentials (MSSE)

By default, WSUS Offline Update downloads these files in the two most often used languages, German and English. Therefore, the supposedly global file StaticDownloadLinks-dotnet-x64-glb.txt just contains one German language pack. Other languages can be added as *custom languages*.

Handling these languages requires at least four additional scripts:

```
AddCustomLanguageSupport.cmd
RemoveCustomLanguageSupport.cmd
RemoveEnglishLanguageSupport.cmd
RemoveGermanLanguageSupport.cmd
```

But this turned out to be quite complicated for both users and other developers.

A unified approach for language settings

The new Linux scripts use a unified approach:

1. The default languages German and English are removed from the global static download files on the first run.
2. Users must always specify one real language like *deu* or *enu* on the command line; the placeholder *glb* is not allowed for any update.

   Since version 1.0-beta-4, you can also join several languages to a comma-separated list like *deu,enu*.

3. These languages are used wherever a language setting is needed: They are used like the *default* and *custom* languages to include localized downloads for Internet Explorer, .NET Frameworks and Security Essentials. For Office 2007 – 2013, they are used as the *update* languages.
4. This way, users get downloads for the specified languages only, and nothing else.

In the first versions of the Linux download scripts, the setup script `update-generator.bash` did not support multiple selections, nor would the download script `download-updates.bash` recognizes multiple languages on the command-line. This meant, that different languages had to be downloaded in turn, and previous downloads hat to be preserved between runs.

The cleanup function handles this by treating the complete static directory as an additional white list. Technically, this is just a recursive `grep` for the filename. Files, which are not in the current download set, but which can still be found in the static directory, are reported as *valid static files*. These files were never automatically deleted. If they were not needed anymore, they had to be manually deleted.

While the concept of *valid static files* was introduced to preserve localized downloads between runs, the same mechanism also protects some other files:

- If service packs have been downloaded before, and the option `-includesp` is no longer used, the files are still preserved.
- 64-bit Office downloads are not deleted, if only the corresponding 32-bit downloads are selected.

Again, if these files are not needed anymore, they must be manually deleted.

In recent versions of the Linux download scripts, the frontend script update-generator.bash allows multiple selections in all dialogs with the help of the external utility `dialog`, and the script download-updates.bash can download all updates and all languages in one pass.

This removes the need to protect *valid static files*, which was rather a workaround to keep localized updates between different runs. To allow these files to be automatically cleaned up, you can set the option `keep_valid_static_files` to disabled in the preferences file.

Command-line options

The command-line options for the download script, version 2.0 for the current version WSUS Offline Update 12.0 are:

```
download-updates.bash: Download updates for Microsoft Windows and Office

USAGE
   ./download-updates.bash UPDATE[,UPDATE...] LANGUAGE[,LANGUAGE...] \
   [OPTIONS]

UPDATE
```

```
    w62-x64       Windows Server 2012, 64-bit
    w63           Windows 8.1, 32-bit
    w63-x64       Windows 8.1 / Server 2012 R2, 64-bit
    w100          Windows 10, 32-bit
    w100-x64      Windows 10 / Server 2016, 64-bit
    o2k10         Office 2010, 32-bit
    o2k10-x64     Office 2010, 32-bit and 64-bit
    o2k13         Office 2013, 32-bit
    o2k13-x64     Office 2013, 32-bit and 64-bit
    o2k16         Office 2016, 32-bit
    o2k16-x64     Office 2016, 32-bit and 64-bit
    all           All Windows and Office updates, 32-bit and 64-bit
    all-x86       All Windows and Office updates, 32-bit
    all-x64       All Windows and Office updates, 64-bit
    all-win       All Windows updates, 32-bit and 64-bit
    all-win-x86   All Windows updates, 32-bit
    all-win-x64   All Windows updates, 64-bit
    all-ofc       All Office updates, 32-bit and 64-bit
    all-ofc-x86   All Office updates, 32-bit

    Notes: Multiple updates can be joined to a comma-separated list like
    "w63,w63-x64".

LANGUAGE
    deu    German
    enu    English
    ara    Arabic
    chs    Chinese (Simplified)
    cht    Chinese (Traditional)
    csy    Czech
    dan    Danish
    nld    Dutch
    fin    Finnish
    fra    French
    ell    Greek
    heb    Hebrew
    hun    Hungarian
    ita    Italian
    jpn    Japanese
    kor    Korean
    nor    Norwegian
    plk    Polish
    ptg    Portuguese
    ptb    Portuguese (Brazil)
    rus    Russian
    esn    Spanish
    sve    Swedish
    trk    Turkish
```

```
    Note: Multiple languages can be joined to a comma-separated list like
    "deu,enu".

OPTIONS
   -includecpp
        Include Visual C++ runtime libraries

   -includedotnet
        Include .NET Frameworks: localized installation files and updates

   -includewddefs
        Windows Defender definition updates for the built-in Defender
        of  Windows 8, 8.1 and 10.

COMPATIBILITY
    All three options can only be used with Windows updates.

    These optional downloads depend on the architecture of the operating
    system, and that can only be derived from the selected Windows
    versions, or from internal lists, which include Windows updates.

NOTES
    Windows Vista and higher are multilingual, but WSUS Offline Update
    still needs the correct language settings to download:
    - language packs for Internet Explorer 11 on Windows Server 2012,
      for languages other than English
    - language packs for .NET Frameworks

    In the Linux download scripts, all needed languages must be given on
    the command-line. For convenience, you can join several languages
    to a comma-separated list, to allow a faster evaluation of Windows
    Server 2012 and .NET Frameworks.

EXAMPLES
    To get updates for Windows 7 with all optional downloads in German
    and English, you could either use:

    ./download-updates.bash w61 deu -includesp -includecpp \
                -includedotnet -includewddefs -includemsse
    ./download-updates.bash w61 enu -includesp -includecpp \
                -includedotnet -includewddefs -includemsse

    or:

    ./download-updates.bash w61 deu,enu -includesp -includecpp \
                -includedotnet -includewddefs -includemsse

    This should get the same updates as the Windows script
    DownloadUpdates.cmd in its default configuration, using the default
```

```
languages German and English.

To get updates for Windows 8.1 with all optional downloads in French
and Spanish, you could use:

./download-updates.bash w63 fra,esn -includesp -includecpp \
            -includedotnet -includewddefs8

To get the same results with the Windows script DownloadUpdates.cmd,
you need to run four additional scripts first:

RemoveGermanLanguageSupport.cmd
RemoveEnglishLanguageSupport.cmd
AddCustomLanguageSupport.cmd fra
AddCustomLanguageSupport.cmd esn

Since version 1.1 of the Linux download scripts, updates can also
be joined to a comma-separated list. To get all updates for Windows
7 and Windows 8.1, you can use:

./download-updates.bash w61,w61-x64,w63,w63-x64 deu,enu -includesp \
            -includecpp -includedotnet -includewddefs -includemsse

Note, that wddefs8 is a subset of msse, so you don't need to include
both for such lists.

Single updates like w61 and w61-x64 can also be combined with the
internal lists. To get Windows 7 updates and all Office updates,
you could use:

./download-updates.bash w61,w61-x64,all-ofc deu,enu -includesp \
            -includecpp -includedotnet -includewddefs -includemsse

See the script get-all-updates.bash for more examples. This script
may also serve as a template for customization.
```

This description is also available in the file usage.txt and at the top of the download script itself.

ESR versions

There are two branches of WSUS Offline Update, to support old Office and Windows versions, which don’t receive regular updates anymore.

- The branch WSUS Offline Update 9.2.x ESR supported Windows XP, Server 2003 and Vista. Since about December 2019, this version is broken because of internal changes in the WSUS offline scan file wsusscn2.cab.

  The Linux scripts version 1.1 ESR was created to support this first ESR version.

- The new branch WSUS Offline Update 11.9.x ESR is meant to support Windows Server 2008 and Windows 7 / Server 2008 R2.

  It is supported by the Linux scripts version 1.19.1 ESR.

Comparing the results on Windows and Linux

Selecting the same options in Windows and Linux should result in the same downloads. Optimally, files should not just be compared by their name, but also by their content. This would take a long time for two directories of about 41 GB each. But fortunately, most of this work has already been done by creating the *integrity database* of hashdeep files in the `client/md` directory.

- Each hashdeep file corresponds to one download directory.
- Each line in a hashdeep file is a fingerprint of one downloaded file. It consists of the file size, MD5, SHA-1 and SHA-256 hashes, and the relative file path.

Thus, comparing two directories of small text files is enough for a deep comparison of all downloaded files. This can be easily done with `diff`. Comparing the hashes files also ensures, that these files are in the correct format to be used by the Windows script `DoUpdate.cmd` during installation.

The script `compare-integrity-database.bash` in the directory `comparison-linux-windows` is meant for this comparison. The file `example-results-md.txt` shows typical results: The four virus definition files are usually different, because they change every two hours, but the other files should be the same.

The script `compare-update-tables.bash` does a similar comparison of the `*.csv` files, which are used for the installation of Office updates.

Validation of downloaded files

There are at least three different approaches to validate downloaded files:

Comparing file hashes

For all security updates extracted from the WSUS catalog file `wsusscn2.cab`, the SHA-1 hash is embedded into the filename. It can be easily compared to the calculated hash of the file.

Unfortunately, this doesn’t work for the WSUS catalog file itself and for the virus definition files. But these files create most problems.

Testing the file integrity with cabextract

The integrity of the WSUS catalog file `wsusscn2.cab` is tested with `cabextract -t`. This ensures, that all files in the archive can be expanded. This is not necessary for the other cab archives, because they can be tested by using the SHA-1 hash, which is embedded in the file name, as the reference value.

The virus definition files are provided as .exe files, but they are basically self-extracting cab archives. Therefore, they can also be checked with `cabextract -t`.

Validating digital file signatures with Sysinternals Sigcheck

Sysinternals Sigcheck does run under `wine`, but there are a few drawbacks:

- The built-in wine library `CRYPT32.dll` doesn't seem to provide the functionality to really validate file signatures. With this library, Sigcheck can only tell, if a file is *signed* or *unsigned*. That much actually works, but it is not enough to detect subtle problems with the downloaded files. If a file is digitally signed, but the file is damaged for some reason, it is still reported as *signed*. The correct result should be that the signature could not be verified.
- The utility `winetricks` can replace the built-in wine library with a native Windows library. But without the necessary root certificates and complete certificate chains, Sigcheck shows a generic error message for every single file.

Thus, although a preliminary implementation for `wine` and Sigcheck exists, it needs more work, especially to transfer the root certificates from Windows to Linux.

Validating digital file signatures with chktrust

The .NET Framework on Windows and the Mono Framework on Linux provide similar utilities for the same tasks.

The .NET Framework on Windows provides a Certificate Manager for the Microsoft Management Console, which can be launched as:

```
mmc.exe certmgr.msc
```

The Certificate Verification tool (chktrust.exe) and the Sign tool (signtool.exe) can be used to verify digital signatures.

- https://msdn.microsoft.com/en-us/library/z045761b%28v=vs.100%29.aspx
- https://msdn.microsoft.com/en-us/library/8s9b9yaz%28v=vs.110%29.aspx

The Mono framework for Linux provides two similar utilities: `certmgr` and `chktrust`, but the command `chktrust` can only verify executable files, not cab archives.

- http://www.mono-project.com/docs/tools+libraries/tools/

Still, this might be the easiest way to transfer certificates from Windows to Linux. Since wine seems to integrate with the Mono framework, these certificates might even work with Sigcheck.

Missing functionality

- Download from local WSUS servers

  Implementing and testing this function would require a WSUS server, which I don't have.

Also, downloading from a local WSUS server may not work as expected: Only *dynamic* download links, which are extracted from the WSUS catalog file wsusscn2.cab, can be redirected to a local WSUS server. *Static* download links are still downloaded from the Microsoft download sites. This includes the catalog file wsusscn2.cab itself and all other download links from the static download files in the wsusoffline/static directory. The domain *wsusoffline.net* will also be contacted to check for updates of the configuration files.

So, the *download part* of WSUS Offline Update does not really work without an Internet connection, and it was never meant to do so. The *installation part* of WSUS Offline Update does work without Internet connections.

Local WSUS servers may still be useful to restrict the downloads to *approved* updates only.

File version

This file was last modified on 2020-03-25.