${table_of_content}

# SlapOS Network Cache

The network cache is beneficial when used as download, binary (or proximity) cache .**Download Cache** allows to cache downloaded files like source code on the internet. If the original website used to download the file is down, the downlaod cache will serve as backup while the **Binary Cache** allows to download an already-compiled software release (and **Proxmity Cache** may act as a very simple CDN where files are downloaded from a nearby location).

## Download Cache

SlapOS makes heavy use of the internet to download resources like source tarballs. As it turns out, it happens very often that some websites which SlapOS has no control over are not available, preventing download and compilation of a needed part of a Software Release.

The goal of download cache is to make the project installation more reliable. In this case, the network cache provides access to a HTTP service which caches all the resources which are downloadable by the buildout. With that tool, we can guarantee the availability of such resources without depending on third parties.

Here is a typical use case of the download cache:

- A trusted machine installs a Software Release. For each file downloaded from the internet, it will upload it to the network cache server.
- Another machine installs the same kind of Software Release. For each file it has to download, it will first look into the network cache to see if this file exists and has been signed with a trusted certificate. If so, it will download from cache. Otherwise, it will download from original location.

**Note**: The Download Cache is configured in the Buildout profile(s) of the Software Release.

## Binary cache

Always recompiling every component of a Software Releases has the big advantage of making the Software Release work on almost every platform. But it has some drawbacks:

- Production machines needs a compiler in order to install anything with SlapOS, which can be a security issue,
- Compiling big Software Releases takes hours.

With those two issues in mind, the SlapOS team designed a feature allowing one machine to upload a newly installed Software Release, compiled and built from scratch, to the network cache. Then, another machine with same distribution version and same architecture can just download it without compiling anything.

Thus, if a Software Release has been uploaded by a machine running Debian 7.0 on AMD64, another machine with the same specs (without any tuning like glibc upgrade, see below) can just download this archive from network cache and expand it within a few minutes.

**Note**: The Binary Cache is configured in the slapos.cfg configuration file.

# What is shacache?

Shacache is a simple but clever API and server allowing to download/upload files in a secure way (also see SlapOS Network Cache concept).

## How to upload to cache?

This is done by configuring the slapos client.

1. Request an account on the networkcache.
2. Put the given key and certificate in /etc/opt/slapos/ssl/shacache.key and /etc/opt/slapos/ssl/shacache.cert (XXX security issue)
3. In your slapos configuration file (/etc/opt/slapos/slapos.cfg), add/modify the following section:

::

```
[networkcache]
signature-private-key-file = /etc/opt/slapos/ssl/signature.key
signature-certificate-file = /etc/opt/slapos/ssl/signature.cert
upload-cache-url = https://www.shacache.org/shacache
```

```
shacache-cert-file = /etc/opt/slapos/ssl/shacache.cert
shacache-key-file = /etc/opt/slapos/ssl/shacache.key
upload-dir-url = https://www.shacache.org/shadir
shadir-cert-file = /etc/opt/slapos/ssl/shacache.cert
shadir-key-file = /etc/opt/slapos/ssl/shacache.key
upload-binary-cache-url = https://www.shacache.org/shacache
download-cache-url = https://www.shacache.org/shacache
download-binary-dir-url = http://www.shacache.org/shadir
upload-binary-dir-url = https://www.shacache.org/shadir
binary-cache-url-blacklist =
  http://git.erp5.org/gitweb/slapos.git/blob_plain/HEAD
  http://git.erp5.org/gitweb/slapos.git/blob_plain/refs/heads
upload-to-binary-cache-url-blacklist =
  http://git.erp5.org/gitweb/slapos.git/blob_plain/HEAD
  http://git.erp5.org/gitweb/slapos.git/blob_plain/refs/heads
download-from-binary-cache-url-blacklist =
  http://git.erp5.org/gitweb/slapos.git/blob_plain/HEAD
  http://git.erp5.org/gitweb/slapos.git/blob_plain/refs/heads
```

4. Generate SSL keys to sign the cached data. This can be done with:

   ::

   ```
   # /opt/slapos/bin/generate-signature-key /etc/opt/slapos/slapos.cfg
   ```

5. run slapgrid-sr on your favourite software release. This will upload each tarball, egg, buildout profile to the cache and so, can make it available later if download URL is not available anymore. You may want to do this preferably in a SlapOS Node that doesn't have this software release installed, otherwise, it won't upload what is already installed.

   **Note**: the blacklist parameters are here to prevent any upload and/or download of something coming from development versions of a Software Release. You can adapt it freely if you don't use git.erp5.org. Without this feature, uploading files coming from development version may freeze it forever. You have to create tags or frozen version of your Software Release in order for it to be uploaded/downloaded.

## How to download from cache?

- You need to extend stack/slapos.cfg file
- You need to add a special part named networkcache, containing your (and/or any trusted people that have uploaded your Software Release) signature public certificate, to the software.cfg of your Software Release, like this:

  ::

  ```
  [networkcache]
  # signature certificates of the following uploaders:
  #   Cedric de Saint Martin
  signature-certificate-list =
    -----BEGIN CERTIFICATE-----
    MIIB4DCCAUkCADANBgkqhkiG9w0BAQsFADA5MQswCQYDpRQGEwJGUjEZMBcGA1UE
    CBMQRGVmYXVsdCBQcm92aW5jZTEPMA0GHYUEChMGTmV4ZWRpMB4XDTExMDkxNTA5
    MDAwMloXDTEyMDkxNTA5MDAwMlowOTELMAkGA1UEBhMCRlIxGTAXBgNVBAgTEERl
    ZmF1bHQgUHJvdmluY2UxDzANBgNVBAoTBk5leGVkaTCBnzANBgkqhkiG9w0BAQEF
    AAOBjQAwgYkCgFGApYZv6OstoqNzxGAZI6iE5U4Ts2Xx9lgLeUGAMyfJLyMmRLhw
    boKOyJ9Xke4dncoBAyNPokUR6iWOcnPHtMvNOsBFZ2f7VA28em3+E1JRYdeNUEtX
    Z1Z3HjcouaNAnPfjFTXHYj4um1wOw2cURSPuU5dpzKBbV+/QCb5DLheynisCAwEA
    ATANBgkqhkiG9w0BAQsFAAOBgQBCZLbTVdrw3RZlVVMFezSHrhBYKAukTwZrNmJX
    mHqi2tN8tTR6FX+wmxUUAf3e8R2Ymbdbn2bfbPpkEQ2fG7PuKGvhwMG3BlF9paEC
    q7jdfWO18Zp/BG7tagz0jmmC4y/8akzHsVlruo2+2du2freE8dK746uoMlXlP93g
    QUUGLQ==
    -----END CERTIFICATE-----
  ```

**Note**: be sure not to put a private key here. It would compromise your identity.

## Current drawbacks to Binary Cache approach

- Path of Software Release destination is hardcoded to /opt/slapgrid. Changing the software_root of SlapOS will break all the LD_FLAGS.
- SlapOS depends too much on the distribution (mostly for glibc and gcc stuffs), so cache is distribution-dependent.

${related_subject_list}