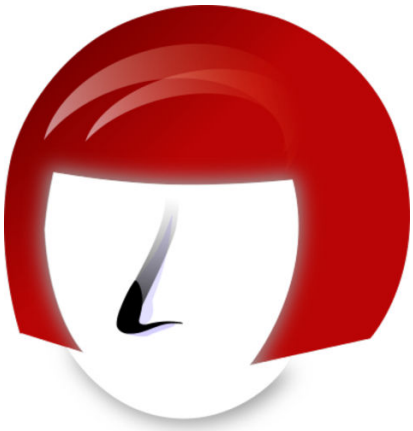
The Kleopatra Handbook

Marc Mutz
Developer: David Faure
Developer: Steffen Hansen
Developer: Matthias Kalle Dalheimer
Developer: Jesper Pedersen
Developer: Daniel Molkentin



Contents

List of Tables

Abstract

Kleopatra is a tool for managing X.509 and OpenPGP certificates.

Chapter 1

Introduction

Kleopatra is the KDE tool for managing X.509 and OpenPGP certificates in the GpgSM and GPG keyboxes and for retrieving certificates from LDAP and other certificate servers.

Kleopatra can be started from KMail's **Tools** → **Certificate Manager** menu, as well as from the command line. The Kleopatra executable is named **`kleopatra`**.

> NOTE
> This program is named after Cleopatra, a famous female Egyptian pharaoh that lived at the time of Julius Caesar, with whom she had a child, Caesarion, unacknowledged as his heir.
> The name was chosen since this program originates from the Ägypten Projects (Ägypten is German for Egypt). Kleopatra is the German spelling of Cleopatra.

Chapter 2

Main Functions

2.1 Viewing the Local Keybox

Kleopatra's main function is to display and edit the contents of the local keybox, which is similar to GPG's concept of keyrings, albeit one should not stretch this analogy too much.

The main window is divided into the large key listing area consisting of several tabs, the menubar and the search bar on top, and a status bar at the bottom.

Each line in the key list corresponds to one certificate, identified by the so-called **Subject DN**. DN is an acronym for 'Distinguished Name', a hierarchical identifier, much like a file system path with an unusual syntax, that is supposed to globally uniquely identify a given certificate.

To be valid, and thus usable, (public) keys need to be signed by a CA (Certification Authority). These signatures are called certificates, but usually the terms 'certificate' and '(public) key' are used interchangeably, and we will not distinguish between them in this manual either, except when explicitly noted.

CAs must in turn be signed by other CAs to be valid. Of course, this must end somewhere, so the top-level CA (root-CA) signs its key with itself (this is called a self-signature). Root certificates thus need to be assigned validity (commonly called trust) manually, e.g. after comparing the fingerprint with the one on the website of the CA. This is typically done by the system administrator or the vendor of a product using certificates, but can be done by the user via GpgSM's command line interface.

To see which of the certificates are root certificates, you switch to the hierarchical keylist mode with **View → Hierarchical Certificate List**.

You can see the details of any certificate by double-clicking it or using **View → Certificate Details**. This opens a dialog that shows the most common properties of the certificate, its certificate chain (i.e. the chain of issuers up to the root-CA), and a dump of all information the backend is able to extract from the certificate.

If you change the keybox without using Kleopatra (e.g. using GpgSM's command line interface), you can refresh the view with **View → Redisplay (F5)** .

2.2 Searching and Importing Certificates

Most of the time, you will acquire new certificates by verifying signatures in emails, since certificates are embedded in the signatures made using them most of the time. However, if you need to send a mail to someone you have not yet had contact with, you need to fetch the certificate from an LDAP folder (although GpgSM can do this automatically), or from a file. You also need to import your own certificate after receiving the CA answer to your certification request.

To search for a certificate in an LDAP directory, select **File** → **Lookup Certificates on Server** and enter some text (e.g. the name of the person you want the certificate for) into the line edit of the **Keyserver Certificate Lookup** dialog, then click on the **Search** button. The results will be displayed in the key list below the search bar, where you can select certificates to look at them by clicking the **Details** button or download them with **Import** into the local keybox.

You can configure the list of LDAP servers to search in the **Directory Services** page of Kleopatra's configure dialog.

If you received the certificate as a file, try **File** → **Import Certificates... (Ctrl+I)** . GpgSM needs to understand the format of the certificate file; please refer to GpgSM's manual for a list of supported file formats.

If you did not create your keypair with GpgSM, you also need to manually import the public key (as well as the secret key) from the PKCS#12 file you got from the CA. You can do this on the command line with `kleopatra --import-certificate filename` or from within Kleopatra with **File** → **Import Certificates... (Ctrl+I)** , just as you would to for 'normal' certificates.

2.3 Creating New Key Pairs

The menu item **File** → **New Certificate... (Ctrl+N)** starts the **Key Pair Creation Wizard** which will guide you through a number of steps to create a certificate request.

Whenever you are done with a step in the wizard, press **Next** to go to the next step (or **Back** to review steps that are already completed). The certificate request creation can be canceled at any time by pressing the **Cancel** button.

On the first page of the wizard choose which type of certificate you want to create:

Create a personal OpenPGP key pair

OpenPGP key pairs are created locally, and certified by your friends and acquaintances. There is no central certification authority; instead, every individual creates a personal Web Of Trust by certifying other user's key pairs with his own certificate.

You have to enter a **Name**, **EMail** and optional a **Comment**.

Create a personal X.509 key pair and certification request

X.509 key pairs are created locally, but certified centrally by a certification authority (CA). CAs can certify other CAs, creating a central, hierarchical chain of trust.

The next step in the wizard is to type in your personal data for the certificate. The fields to fill out are:

- **Common Name (CN):** Your name;
- **Email address (EMAIL):** Your email address; be sure to type this in correctly—this will be the address people will be sending mail to when they use your certificate.
- **Location (L):** The town or city in which you live;
- **Organizational unit (OU):** The organizational unit you are in (for example, ˝Logistics˝);
- **Organization (O):** The organization you represent (for example, the company you work for);
- **Country code (C):** The two letter code for the country in which you are living (for example, ˝US˝);

The next step in the wizard is to select whether to store the certificate in a file or send it directly to a CA. You will have to specify the filename or email address to send the certificate request to.

2.3.1 Revoking a key

A key pair that has expired can be brought back into an operational state as long as you have access to the private key and the passphrase. To reliably render a key unusable you need to revoke it. Revoking is done by adding a special revocation signature to the key.

This revocation signature is stored in a separate file. This file can later be imported into the keyring and is then attached to the key rendering it unusable. Please note that to import this signature to the key no password is required. Therefore you should store this revocation signature in a safe place, usually one that is different from you key pair. It is a good advise to use a place that is detached from your computer, either copy it to an external storage device like an USB stick or print it out.

Kleopatra does not provide a function to create such a revocation signature at any time, but you can do that with the KDE application KGpg by choosing **Keys** → **Revoke key** and optionally importing the revocation signature to your keyring immediately.

An alternative way of generating a revocation certificate is to use GPG directly from the command line: **`gpg --output revocation_certificate.asc --gen-revoke your_key`**. The argument *`your_key`* must be a key specifier, either the key ID of your primary keypair or any part of a user ID that identifies your keypair.

Chapter 3

Menu Reference

3.1 File Menu

File → New Certificate... (Ctrl+N)

Creates a new key pair (public and private) and allows to send the public part to a certification authority (CA) for signing. The resulting certificate is then sent back to you, or stored in an LDAP server for you to download into your local keybox, where you can use it to sign and decrypt mails.

This mode of operation is called 'decentralized key generation', since all keys are created locally. Kleopatra (and GpgSM) do not support 'centralized key generation' directly, but you can import the public/secret key bundle that you receive from the CA in PKCS#12 format via **File → Import Certificates... (Ctrl+I)** .

File → Lookup Certificates on Server... (Ctrl+Shift+I)

Searches for, and imports, certificates from certificate servers into the local keybox. See Section 2.2 for details.

You need to have key servers configured for this to work. See Section 5.1 for more details.

File → Import Certificates... (Ctrl+I)

Imports certificates and/or secret keys from files into the local keybox. See Section 2.2 for details.

The format of the certificate file must be supported by GpgSM/GPG. Please refer to the GpgSM and GPG manuals for a list of supported formats.

File → Export Certificates... (Ctrl+E)

Exports the selected certificates to a file.

The filename extension you choose for the export file name determines the format of the export file:

- For OpenPGP certificates, `gpg` and `pgp` will result in a binary file, whereas `asc` will result in an ASCII-armored file.
- For S/MIME certificates, `der` will result in a binary, DER-encoded file, whereas `pem` will result in an ASCII-armored file.

Unless multiple certificates are selected, Kleopatra will propose `fingerprint.{asc,pem}` as the export file name.

This function is only available when one or more certificates have been selected.

> **NOTE**
> This function exports only the public keys, even if the secret key is available. Use **File → Export Secret Keys...** to export the secret keys into a file.

File → Export Secret Keys...

Exports the secret key to a file.

In the dialog that opens, you can choose whether to create a binary or an ASCII-armored export file (**ASCII armor**). Next click on the folder icon at the right hand side of the **Output file** text box and select folder and name of the export file. When exporting S/MIME secret keys, you can also choose the **Passphrase charset**. See the discussion of the `--p12-charset charset` option in the GpgSM manual for more details.

This function is only available when exactly one certificate has been selected, and the secret key for that certificate is available.

> **WARNING**
> It should rarely be necessary to use this function, and if it is, it should be carefully planned. Planning the migration of a secret key involves choice of transport media and secure deletion of the key data on the old machine, as well as on the transport medium, among other things.

File → Export Certificates to Server... (Ctrl+Shift+E)

Publish the selected certificates on a keyserver (OpenPGP only).

The certificate is sent to the certificate server configured for OpenPGP (cf. Section 5.1), if that is set, otherwise to `keys.gnupg.net`.

This function is only available if at least one OpenPGP (and no S/MIME) certificates have been selected.

> **NOTE**
> When OpenPGP certificates have been exported to a public directory server, it is nearly impossible to remove them again. Before exporting your certificate to a public directory server, make sure that you have created a revocation certificate so you can revoke the certificate if needed later.

> **NOTE**
> Most public OpenPGP certificate servers synchronize certificates amongst each other, so there is little point in sending to more than one.
> It can happen that a search on a certificate server turns up no results even though you just have sent your certificate there. This is because most public keyserver addresses use DNS round-robin to balance the load over multiple machines. These machines synchronize with each other, but usually only every 24 hours or so.

File → Decrypt/Verify Files...

Decrypts files and/or verifies signatures over files.

File → Sign/Encrypt Files...

Signs and/or encrypts files.

File → Close (Ctrl+W)

Closes Kleopatra's main window. You can restore it from the system tray icon at any time.

File → Quit (Ctrl+Q)

Terminates Kleopatra.

3.2 View Menu

View → Redisplay (F5)

Refreshes the certificate list.
Using this function is usually not necessary, as Kleopatra monitors the file system for changes and automatically refreshes the certificate list when needed.

View → Stop Operation (Esc)

Stops (cancels) all pending operations, e.g. a search, keylisting, or a download.

This function is only available if at least one operation is active.

> NOTE
> Due to backend limitations, sometimes operations will hang in such a way that this function won't be able to cancel them, right away, or at all.
> In such cases, the only way to restore order is to kill SCDaemon, DirMngr, GpgSM and GPG processes, in that order, via the operating system tools (**top**, Task-Manager, etc.), until the operation get unblocked.

View → Certificate Details

Shows the details of the currently selected certificate.

This function is only available if exactly one certificate is selected.

This function is also available by double-clicking the corresponding item in the list view directly.

View → Hierarchical Certificate List

Toggles between hierarchical and flat certificate list mode.

In hierarchical mode, certificates are arranged in issuer/subject relation, so it is easy to see which certification hierarchy a given certificate belongs to, but a given certificate is harder to find initially (though you can of course use the search bar).

In flat mode, all certificates are displayed in a flat list, sorted alphabetically. In this mode, a given certificate is easy to find, but it is not directly clear which root certificate it belongs to.
This function toggles hierarchical mode per tab, i.e. each tab has its own hierarchy state. This is so that you can have both a flat and a hierarchical listing at hand, each in its own tab.

> NOTE
> Hierarchical display is currently only implemented for S/MIME certificates. There is disagreement amongst the developers regarding the correct way to display OpenPGP certificates hierarchically (basically, 'parent = signer' or 'parent = signee').

View → Expand All (Ctrl+.)

Expands all list items in the certificate list view, i.e. makes all items visible.

This is the default when entering hierarchical keylist mode.

You can still expand and collapse each individual item by itself, of course.

This function is only available when **View → Hierarchical Certificate List** is on.

View → Collapse All (Ctrl+,)

Collapses all list items in the certificate list view, i.e. hides all but the top-level items.

You can still expand and collapse each individual item by itself, of course.

This function is only available when **View → Hierarchical Certificate List** is on.

3.3 Certificates Menu

Certificates → Change Owner Trust...

Changes the Owner Trust of the selected OpenPGP certificate.

This function is only available when exactly one OpenPGP certificate is selected.

Certificates → Trust Root Certificate

Marks this (S/MIME) root certificate as trusted.

In some ways, this is the equivalent of **Certificates → Change Owner Trust...** for S/MIME root certificates. You can, however, only choose between—in OpenPGP terms—'ultimate' trust and 'never trust'.

> NOTE
> The backend (by way of GpgAgent) will ask at root certificate import time whether to trust the imported root certificate. However, that function must be explicitly enabled in the backend configuration (`allow-mark-trusted` in `gpg-agent.conf`, or either **GnuPG System → GPG Agent → Allow clients to mark keys as "trusted"** or **S/MIME Validation → Allow to mark root certificates as trusted** under chapter 5).
> Enabling that functionality in the backend can lead to popups from PinEntry at inopportune times (e.g. when verifying signatures), and can thus block unattended email processing. For that reason, and because it is desirable to be able to *distrust* a trusted root certificate again, Kleopatra allows manual setting of trust.

> WARNING
> Due to lack of backend support for this function, Kleopatra needs to work directly on the GpgSM trust database (`trustlist.txt`). When using this function, make sure no other crypto operations are in progress that could race with Kleopatra for modifications to that database.

This function is only available when exactly one S/MIME root certificate is selected, and that certificate is not yet trusted.

Use **Certificates → Distrust Root Certificate** to undo this function.

Certificates → Distrust Root Certificate

Marks this (S/MIME) root certificate as not trusted.

This function is only available when exactly one S/MIME root certificate is selected, and that certificate is currently trusted.

Used to undo **Certificates → Trust Root Certificate** . See there for details.

Certificates → Certify Certificate...

Allows you to certify another OpenPGP certificate.

This function is only available if exactly one OpenPGP certificate is selected.

Certificates → Change Expiry Date...

Allows to change the expiry date of your OpenPGP certificate.

Use this function to extend the lifetime of your OpenPGP certificates as an alternative to either creating a new one, or using unlimited lifetime ('never expires').

This function is only available if exactly one OpenPGP certificate is selected, and the secret key is available for that certificate.

Certificates → Change Passphrase...

Allows to change the passphrase of your secret key.

This function is only available if exactly one certificate is selected, and the secret key is available for that certificate. It requires a very recent backend, since we changed the implementation from direct calling of GPG and GpgSM to a GpgME-based one.

> NOTE
> For security reasons, both the old as well as the new passphrase is asked for by PinEntry, a separate process. Depending on the platform you are running on and on the quality of the PinEntry implementation on that platform, it may happen that the PinEntry window comes up in the background. So, if you select this function and nothing happens, check the operating system's task bar in case a PinEntry window is open in the background.

Certificates → Add User-ID...

Allows to add a new User-ID to your OpenPGP certificate.

Use this to add new identities to an existing certificate as an alternative to creating a new key pair. An OpenPGP user-ID has the following form:

```
Real Name (Comment) <Email>
```

In the dialog that comes up when you select this function, Kleopatra will ask you for each of the three parameters (*Real Name*, *Comment* and *Email*) separately, and display the result in a preview.

> NOTE
> These parameters are subject to the same Administrator restrictions as in new certificates. See Section 2.3 and Section 6.1 for details.

This function is only available when exactly one OpenPGP certificate is selected, and the secret key is available for that certificate.

Certificates → Delete (Del)

Deletes the selected certificates from the local keyring.

Use this function to remove unused keys from your local keybox. However, since certificates are typically attached to signed emails, verifying an email might result in the key just removed to pop back into the local keybox. So it is probably best to avoid using this function as much as possible. When you are lost, use the search bar or the **View → Hierarchical Certificate List** function to regain control over the lot of certificates.

> WARNING
> There is one exception to the above: When you delete one of your own certificates, you delete the secret key along with it. This implies that you will not be able to read past communication encrypted to you using this certificate, unless you have a backup somewhere.
> Kleopatra will warn you when you attempt to delete a secret key.

Due to the hierarchical nature of S/MIME certificates, if you delete an S/MIME issuer certificate (CA certificate), all subjects are deleted, too.[1]

Naturally, this function is only available if you selected at least one certificate.

Certificates → Dump Certificate

Shows all information that GpgSM has about the selected (S/MIME) certificate.

See the discussion about `--dump-key key` in the GpgSM manual for details about the output.

[1] This is the same as a filesystem: When you delete a folder, you delete all files and folders in it, too.

3.4 Tools Menu

Tools → GnuPG Log Viewer...

Starts KWatchGnuPG, a tool to present the debug output of GnuPG applications. If signing, encryption, or verification mysteriously stop working, you might find out why by looking at the log.

This function is not available on Windows®, since the underlying mechanisms are not implemented in the backend on that platform.

Tools → Refresh OpenPGP Certificates

Refreshes all OpenPGP certificates by executing

```
gpg --refresh-keys
```

After successful completion of the command, your local keystore will reflect the latest changes with respect to validity of OpenPGP certificates.

See note under **Tools → Refresh X.509 Certificates** for some caveats.

Tools → Refresh X.509 Certificates

Refreshes all S/MIME certificates by executing

```
gpgsm -k --with-validation --force-crl-refresh --enable-crl-checks
```

After successful completion of the command, your local keystore will reflect the latest changes with respect to validity of S/MIME certificates.

> NOTE
> Refreshing X.509 or OpenPGP certificates implies downloading all certificates and CRLs, to check if any of them have been revoked in the meantime.
> This can put a severe strain on your own as well as other people's network connections, and can take up to an hour or more to complete, depending on your network connection, and the number of certificates to check.

Tools → Import CRL From File...

Lets you manually import CRLs from files.

Normally, Certificate Revocation Lists (CRLs) are handled transparently by the backend, but it can sometimes be useful to import a CRL manually into the local CRL cache.

> NOTE
> For CRL import to work, the DirMngr tool must be in the search `PATH`. If this menu item is disabled, you should contact the system administrator and ask them to install DirMngr.

Tools → Clear CRL Cache

Clears the GpgSM CRL cache.

You probably never need this. You can force a refresh of the CRL cache by selecting all certificates and using **Tools → Refresh X.509 Certificates** instead.

Tools → Dump CRL Cache

Shows the detailed contents of the GpgSM CRL cache.

3.5 Settings Menu

Kleopatra has a default KDE **Settings** menu as described in the KDE Fundamentals with one additional entry:

Settings → **Perform Self-Test**

Performs a set of self-tests and presents their result.

This is the same set of tests that is run at startup by default. If you disabled startup-time self-tests, you can re-enable them here.

3.6 Window Menu

The **Window** menu allows you to manage the tabs. Using the items in this menu you can rename a tab, add a new tab, duplicate the current tab, close the current tab, and move the current tab to the left or right.

By clicking with the right mouse button click on a tab you open a context menu, where you can also select the same actions.

3.7 Help Menu

Kleopatra has a default KDE **Help** menu as described in the KDE Fundamentals.

Chapter 4

Command Line Options Reference

Only the options specific to Kleopatra are listed here. As with all KDE applications, you can get a complete list of options by issuing the command **`kleopatra --help`**.

`--uiserver-socket` ***`argument`***
: Location of the socket the ui server is listening on

`--daemon`
: Run UI server only, hide main window

`-p --openpgp`
: Use OpenPGP for the following operation

`-c --cms`
: Use CMS (X.509, S/MIME) for the following operation

`-i --import-certificate`
: Specifies a file or URL from which to import certificates (or secret keys) from.
This is the command line equivalent of **File → Import Certificates... (Ctrl+I)** .

`-e --encrypt`
: Encrypt file(s)

`-s --sign`
: Sign file(s)

`-E --encrypt-sign`
: Encrypt and/or sign file(s). Same as `--sign-encrypt`, do not use

`-d --decrypt`
: Decrypt file(s)

`-V --verify`
: Verify file/signature

`-D --decrypt-verify`
: Decrypt and/or verify file(s)

Chapter 5

Configuring Kleopatra

Kleopatra's configure dialog can be accessed via **Settings** → **Configure Kleopatra...**

Each of its pages is described in the sections below.

5.1 Configuring Directory Services

On this page, you can configure which LDAP servers to use for S/MIME certificate searches, and which key servers to use for OpenPGP certificate searches.

> NOTE
> This is simply a more user-friendly version of the same settings you also find in Section 5.5. Everything you can configure here, you can configure there, too.

> A NOTE ON PROXY SETTINGS
> Proxy settings can be configured for HTTP and LDAP in Section 5.4, but only for GpgSM. For GPG, due to the complexity of keyserver options in GPG and lack of proper support for them in GpgConf, you currently need to modify the config file `gpg.conf` directly. Please refer to the GPG manual for details. Kleopatra will preserve such settings, but does not yet allow to modify them in the GUI.

The **Directory services** table shows which servers are currently configured. Double-click on a cell in the table to change parameters of existing server entries.

The meaning of the columns in the table is as follows:

Scheme
: Determines the network protocol which is used to access the server. Often-used schemes include **ldap** (and its SSL-secured sibling **ldaps**) for LDAP servers (common protocol for S/MIME; the only one supported by GpgSM), and **hkp**, the Horowitz Keyserver Protocol, nowadays usually HTTP Keyserver Protocol, a HTTP-based protocol that virtually all public OpenPGP keyservers support.

  Please refer to the GPG and GpgSM manuals for a list of supported schemes.

Server Name
: The domain name of the server, e.g. `keys.gnupg.net`.

Server Port

The network port the server is listening on.

This changes automatically to the default port when you change the **Scheme**, unless it was set to some non-standard port to begin with. If you changed the default port and cannot get it back, try setting **Scheme** to `http` and **Server Port** to `80` (the default for HTTP), then take it from there.

Base DN

The Base-DN (only for LDAP and LDAPS), i.e. the root of the LDAP hierarchy to start from. This is often also called 'search root' or 'search base'.

It usually looks like `c=de,o=Foo`, given as part of the LDAP URL.

User Name

The user name, if any, to use for logging into the server.

This column is only shown if the option **Show user and password information** (below the table) is checked.

Password

The password, if any, to use for logging into the server.

This column is only shown if the option **Show user and password information** (below the table) is checked.

X.509

Check this column if this entry should be used for X.509 (S/MIME) certificate searches.

Only LDAP (and LDAPS) servers are supported for S/MIME.

OpenPGP

Check this column if this entry should be used for OpenPGP certificate searches.

You can configure as many S/MIME (X.509) servers as you want, but only one OpenPGP server is allowed at any time. The GUI will enforce this.

To add a new server, click on the **New** button. This duplicates the selected entry, if any, or else inserts a default OpenPGP server. Then you can set the **Server Name**, the **Server Port**, the **Base DN**, and the usual **Password** and **User Name**, both of which are only needed if the server requires authentication.

To directly insert an entry for X.509 certificates, use **New** → **X.509**; use **New** → **OpenPGP** for OpenPGP.

To remove a server from the search list, select it in the list, then press the **Delete** button.

To set the LDAP timeout, i.e. the maximum time the backend will wait for a server to respond, simply use the corresponding input field labeled **LDAP timeout (minutes:seconds)**.

If one of your servers has a large database, so that even reasonable searches like `Smith` hit the **maximum number of items returned by query**, you might want to increase this limit. You can find out easily if you hit the limit during a search, since a dialog box will pop up in that case, telling you that the results have been truncated.

> NOTE
> Some servers may impose their own limits on the number of items returned from a query. In this case, increasing the limit here will not result in more returned items.

5.2 Configuring Appearance

5.2.1 Configuring Tooltips

In the main certificate list, Kleopatra can show details about a certificate in a tooltip. The information displayed is the same as in the **Overview** tab of the **Certificate Details** dialog. Tooltips, however, can be restricted to show only a subset of information for a less verbose experience.

> NOTE
> The **Key-ID** is *always* shown. This is to ensure that tooltips for different certificates do, in fact, differ (this is especially important if only **Show validity** has been selected).

You can independently enable or disable the following information sets:

Show validity

Shows information about the validity of a certificate: its current status, issuer-DN (S/MIME only), expiry dates (if any) and certificate usage flags.

Example:

```
This certificate is currently valid.
Issuer:            CN=Test-ZS 7,O=Intevation GmbH,C=DE
Validity:          from 25.08.2009 10:42 through 19.10.2010 10:42
Certificate usage: Signing EMails and Files, Encrypting EMails and ↩
    Files
Key-ID:            DC9D9E43
```

Show owner information

Shows information about the owner of the certificate: subject-DN (S/MIME only), user-IDs (including emails addresses) and ownertrust (OpenPGP only).

OpenPGP example:

```
User-ID:           Gpg4winUserA <gpg4winusera@test.hq>
Key-ID:            C6BF6664
Ownertrust:        ultimate
```

S/MIME example:

```
Subject:           CN=Gpg4winTestuserA,OU=Testlab,O=Gpg4win Project,C= ↩
    DE
a.k.a.:            Gpg4winUserA@test.hq
Key-ID:            DC9D9E43
```

Show technical details

Shows technical information about the certificate: serial number (S/MIME only), type, fingerprint and storage location.

Example:

```
Serial Number:     27
Certificate type:  1,024-bit RSA (secret certificate available)
Key-ID:            DC9D9E43
Fingerprint:       854F62EEEBB41BFDD3BE05D124971E09DC9D9E43
Stored:            on this computer
```

5.2.2 Configuring Certificate Categories

Kleopatra allows you to customize the appearance of certificates in the list view. This includes showing a small icon, but you can also influence the foreground (text) and background colors, as well as the font.

Each certificate category in the list is assigned a set of colors, an icon (optional) and a font in which certificates from that category are displayed. The category list also acts as a preview of the settings. Categories can be freely defined by the administrator or the power user, see Section 6.2 in chapter 6.

To set or change the icon of a category, select it in the list, and press the **Set Icon...** button. The standard KDE icon selection dialog will appear where you can select an existing icon from the KDE collection, or load a custom one.

To remove an icon again, you need to press the **Default Appearance** button.

To change the text (i.e. foreground) color of a category, select it in the list, and press the **Set Text Color...** button. The standard KDE color selection dialog will appear where you can select an existing color or create a new one.

Changing the background color is done in the same way, just press **Set Background Color...** instead.

To change the font, you basically have two options:

1. Modify the standard font, used for all list views in KDE.
2. Use a custom font.

The first option has the advantage that the font will follow whichever style you choose KDE-wide, whereas the latter gives you full control over the font to use. The choice is yours.

To use the modified standard font, select the category in the list, and check or uncheck the font modifiers **Italic**, **Bold**, and/or **Strikeout**. You can immediately see the effect on the font in the category list.

To use a custom font, press the **Set Font...** button. The standard KDE font selection dialog will appear where you can select the new font.

> NOTE
> You can still use the font modifiers to change the custom font, just as for modifying the standard font.

To switch back to the standard font, you need to press the **Default Appearance** button.

5.2.3 Configuring DN-Attribute Order

Although DNs are hierarchical, the order of the individual components (called relative DNs (RDNs), or DN attributes) is not defined. The order in which the attributes are shown is thus a matter of personal taste or company policy, which is why it is configurable in Kleopatra.

> NOTE
> This setting does not only apply to Kleopatra, but to all applications using Kleopatra Technology. At the time of this writing, these include KMail, KAddressBook, as well as Kleopatra itself, of course.

This configuration page basically consists of two lists, one for the known attributes (**Available attributes**), and one describing the **Current attribute order**.

Both lists contain entries described by the short form of the attribute (e.g. **CN**) as well as the spelled-out form (**Common Name**).

The **Available attributes** list is always sorted alphabetically, while the **Current attribute order** list's order reflects the configured DN attribute order: the first attribute in the list is also the one displayed first.

Only attributes explicitly listed in the **Current attribute order** list are displayed at all. The rest is hidden by default.

However, if the placeholder entry **_X_** (**All others**) is in the 'current' list, all unlisted attributes (whether known or not), are inserted at the point of **_X_**, in their original relative order.

A small example will help to make this more clear:

Given the DN

> O=KDE, C=US, CN=Dave Devel, X-BAR=foo, OU=Kleopatra, X-FOO=bar,

the default attribute order of 'CN, L, _X_, OU, O, C' will produce the following formatted DN:

> CN=Dave Devel, X-BAR=foo, X-FOO=bar, OU=Kleopatra, O=KDE, C=US

while 'CN, L, OU, O, C' will produce

> CN=Dave Devel, OU=Kleopatra, O=KDE, C=US

To add an attribute to the display order list, select it in the **Available attributes** list, and press the **Add to current attribute order** button.
To remove an attribute from the display order list, select it in the **Current attribute order** list, and press the **Remove from current attribute order** button.

To move an attribute to the beginning (end), select it in the **Current attribute order** list, and press the **Move to top** (**Move to bottom**) button.

To move an attribute up (down) one slot only, select it in the **Current attribute order** list, and press the **Move one up** (**Move one down**) button.

5.3 Configuring Crypto Operations

5.3.1 Configuring EMail Operations

Here you can configure some aspects of the email operations of Kleopatra's UiServer. Currently, you can only configure whether or not to use 'Quick Mode' for signing and encrypting emails, individually.

When 'Quick Mode' is enabled, no dialog is shown when signing (encrypting) emails, respectively, unless there is a conflict that needs manual resolution.

5.3.2 Configuring File Operations

Here you can configure some aspects of the file operations of Kleopatra's UiServer. Currently, you can only choose the checksum program to use for **CHECKSUM_CREATE_FILES**.

Use **Checksum program to use** to choose which of the configured checksum programs should be used when creating checksum files.

When verifying checksums, the program to use is automatically found, based on the names of the checksum files found.

> NOTE
> The administrator and power user can completely define which checksum programs to make available to Kleopatra through so-called 'Checksum Definitions' in the config file. See Section 6.4 in chapter 6 for details.

5.4 Configuring aspects of S/MIME Validation

On this page, you can configure certain aspects of the validation of S/MIME certificates.

> NOTE
> For the most part, this is simply a more user-friendly version of the same settings you also find in Section 5.5. Everything you can configure here, you can configure there, too, with the exception of **Check certificate validity every *N* hours**, which is Kleopatra-specific.

The meaning of the options is as follows:

5.4.1 Configuring interval certificate checking

Check certificate validity every *N* hours

This option enables interval checking of certificate validity. You can also choose the checking interval (in hours). The effect of interval checking is the same as **View** → **Redisplay (F5)** ; there is no provision for interval scheduling of **Tools** → **Refresh OpenPGP Certificates** or **Tools** → **Refresh X.509 Certificates** .

> NOTE
> Validation is performed implicitly whenever significant files in `~/.gnupg` change. This option, just like **Tools** → **Refresh OpenPGP Certificates** and **Tools** → **Refresh X.509 Certificates** , therefore only affects external factors of certificate validity.

5.4.2 Configuring validation method

Validate certificates using CRLs

If this option is selected, S/MIME certificates are validated using Certificate Revocation Lists (CRLs).

See **Validate certificates online (OCSP)** for alternative method of certificate validity checking.

Validate certificates online (OCSP)

If this option is selected, S/MIME certificates are validated online using the Online Certificates Status Protocol (OCSP).

> WARNING
> When choosing this method, a request is sent to the server of the CA more or less each time you send or receive a cryptographic message, thus theoretically allowing the certificate issuing agency to track whom you exchange (e.g.) mails with.

To use this method, you need to enter the URL of the OCSP responder into **OCSP responder URL**.
See **Validate certificates online (OCSP)** for a more traditional method of certificate validity checking that does not leak information about whom you exchange messages with.

OCSP responder URL

Enter here the address of the server for online validation of certificates (OCSP responder). The URL usually starts with `http://` .

OCSP responder signature

Choose here the certificate with which the OCSP server signs its replies.

Ignore service URL of certificates

Each S/MIME certificate usually contains the URL of its issuer's OCSP responder (**Certificates → Dump Certificate** will reveal whether a given certificate contains it).

Checking this option makes GpgSM ignore those URLs and only use the one configured above.
Use this to e.g. enforce use of a company-wide OCSP proxy.

5.4.3 Configuring validation options

Do not check certificate policies

By default, GpgSM uses the file `~/.gnupg/policies.txt` to check if a certificate policy is allowed. If this option is selected, policies are not checked.

Never consult a CRL

If this option is checked, Certificate Revocation Lists are never used to validate S/MIME certificates.

Allow to mark root certificates as trusted

If this option is checked while a root CA certificate is being imported, you will be asked to confirm its fingerprint and to state whether or not you consider this root certificate to be trusted.
A root certificate needs to be trusted before the certificates it certified become trusted, but lightly allowing trusted root certificates into your certificate store will undermine the security of the system.

> NOTE
> Enabling this functionality in the backend can lead to popups from PinEntry at inopportune times (e.g. when verifying signatures), and can thus block unattended email processing. For that reason, and because it is desirable to be able to *distrust* a trusted root certificate again, Kleopatra allows manual setting of trust using **Certificates → Trust Root Certificate** and **Certificates → Distrust Root Certificate** .
> This setting here does not influence the Kleopatra function.

Fetch missing issuer certificates

If this option is checked, missing issuer certificates are fetched when necessary (this applies to both validation methods, CRLs and OCSP).

5.4.4 Configuring HTTP request options

Do not perform any HTTP requests

Entirely disables the use of HTTP for S/MIME.

Ignore HTTP CRL distribution point of certificates

When looking for the location of a CRL, the to-be-tested certificate usually contains what are known as 'CRL Distribution Point' (DP) entries, which are URLs describing the way to access the CRL. The first-found DP entry is used.

With this option, all entries using the HTTP scheme are ignored when looking for a suitable DP.

Use system HTTP proxy

If this option is selected, the value of the HTTP proxy shown on the right (which comes from the environment variable `http_proxy`) will be used for any HTTP request.

Use this proxy for HTTP requests

If no system proxy is set, or you need to use a different proxy for GpgSM, you can enter its location here.
It will be used for all HTTP requests relating to S/MIME.

The syntax is **`host:port`**, e.g. **`myproxy.nowhere.com:3128`**.

5.4.5 Configuring LDAP request options

Do not perform any LDAP requests

Entirely disables the use of LDAP for S/MIME.

Ignore LDAP CRL distribution point of certificates

When looking for the location of a CRL, the to-be-tested certificate usually contains what are known as ˝CRL Distribution Point˝ (DP) entries, which are URLs describing the way to access the CRL. The first found DP entry is used.

With this option, all entries using the LDAP scheme are ignored when looking for a suitable DP.

Primary host for LDAP requests

Entering an LDAP server here will make all LDAP requests go to that server first. More precisely, this setting overrides any specified *`host`* and *`port`* part in an LDAP URL and will also be used if *`host`* and *`port`* have been omitted from the URL.

Other LDAP servers will be used only if the connection to the 'proxy' failed. The syntax is **`host`** or **`host:port`**. If *`port`* is omitted, port 389 (standard LDAP port) is used.

5.5 Configuring the GnuPG System

This part of the dialog is auto-generated from the output of **gpgconf `--list-components`** and, for each *`component`* that the above command returns, the output of **gpgconf `--list-options` *`component`***.

> **NOTE**
> The most useful of these options have been duplicated as separate pages in the Kleopatra config dialog. See Section 5.1 and Section 5.4 for the two dialog pages which contain selected options from this part of the dialog.

The exact content of this part of the dialog depends on the version of the GnuPG backend you have installed and, potentially, the platform you run on. Thus, we will only discuss the general layout of the dialog, including the mapping from GpgConf option to Kleopatra GUI control.

GpgConf returns configuration information for multiple components. Inside each component, individual options are combined into groups.

Kleopatra displays one tab per component reported by GpgConf; groups are headed by a horizontal line displaying the group name as returned from GpgConf.

Each GpgConf option has a type. Except for certain well-known options which Kleopatra backs with specialised controls for a better user experience, the mapping between GpgConf types and Kleopatra controls is as follows:

GpgConf type	**Kleopatra control**	
	for lists	**for non-lists**
`none`	Spinbox ('count'-semantics)	Checkbox
`string`	N/A	Lineedit
`int32`	Lineedit (unformatted)	Spinbox
`uint32`		
`pathname`	N/A	specialised control
`ldap server`	specialised control	N/A
`key fingerprint`	N/A	
`pub key`		
`sec key`		
`alias list`		

Table 5.1: Mapping From GpgConf Types To GUI Controls

See the GpgConf manual for more information about what you can configure here, and how.

Chapter 6

Administrator's Guide

This Administrator's Guide describes ways to customize Kleopatra that are not accessible via the GUI, but only via config files.

It is assumed that the reader is familiar with the technology used for KDE application configuration, including layout, file system location and cascading of KDE config files, as well as the KIOSK framework.

6.1 Customization of the Certificate-Creation Wizard

6.1.1 Customizing the DN fields

Kleopatra allows you to customize the fields that the user is allowed to enter in order to create their certificate.

Create a group called `CertificateCreationWizard` in the system-wide `kleopatrarc`. If you want a custom order of attributes or if you only want certain items to appear, create a key called `DNAttributeOrder`. The argument is one or more of `CN`,`SN`,`GN`,`L`,`T`,`OU`,`O`,`PC`,`C`,`SP`,`DC`,`BC`,`EMAIL` If you want to initialize fields with a certain value, write something like Attribute=value. If you want the attribute to be treated as a required one, append an exclamation mark (e.g. `CN!`,`L`,`OU`,`O!`,`C!`, `EMAIL!`, which happens to be the default configuration).

Using the KIOSK mode modifier `$e` allows to retrieve the values from environment variables or from an evaluated script or binary. If you want to disallow editing of the respective field in addition, use the modifier `$i`. If you want to disallow the use **Insert My Address** button, set `ShowSetWhoAmI` to false.

> **TIP**
> Due to the nature of the KDE KIOSK framework, using the immutable flag (`$i`) makes it impossible for the user to override the flag. This is intended behavior. `$i` and `$e` can be used with all other config keys in KDE applications as well.

The following example outlines possible customizations:

```
[CertificateCreationWizard]
;Disallow to copy personal data from the addressbook, do not allow local override
ShowSetWhoAmI[$i]=false

;sets the user name to $USER
```

```
CN[$e]=$USER

;sets the company name to "My Company", disallows editing
O[$i]=My Company

;sets the department name to a value returned by a script
OU[$ei]=$(lookup_dept_from_ip)

; sets country to DE, but allows for changes by the user
C=DE
```

6.1.2 Restricting the Types of Keys a User is Allowed to Create

Kleopatra also allows to restrict which type of certificates a user is allowed to create. Note, however, that an easy way around these restrictions is to just create one on the command line.

6.1.2.1 Public Key Algorithms

To restrict the public key algorithm to use, add the config key `PGPKeyType` (and `CMSKeyType`, but only RSA is supported for CMS anyway) to the `CertificateCreationWizard` section of `kleopatrarc`.

The allowed values are `RSA` for RSA keys, `DAS` for DSA (sign-only) keys, and `DSA+ELG` for a DSA (sign-only) key with an Elgamal subkey for encryption.

The default is read from GpgConf or else `RSA` if GpgConf doesn't provide a default.

6.1.2.2 Public Key Size

To restrict the available keys sizes for a public algorithm, add the config key `<ALG>KeySizes` (where *ALG* may be `RSA`, `DSA` or `ELG`) to the `CertificateCreationWizard` section of `kleopatrarc`, containing a comma-separated list of keysizes (in bits). A default may be indicated by prefixing the keysize with a hyphen (-).

```
RSAKeySizes = 1536,-2048,3072
```

The above would restrict allowed RSA key sizes to 1536, 2048 and 3072, with 2048 the default.

In addition to the sizes themselves, you may also specify labels for each of the sizes. Simply set the config key `ALGKeySizeLabels` to a comma-separated list of labels.

```
RSAKeySizeLabels = weak,normal,strong
```

The above, in connection with the previous example, would print something like the following options for selection:

```
weak (1536 bits)
normal (2048 bits)
strong (3072 bits)
```

The defaults are as if the following was in effect:

```
RSAKeySizes = 1536,-2048,3072,4096
RSAKeySizeLabels =
DSAKeySizes = -1024,2048
DSAKeySizeLabels = v1,v2
ELGKeySizes = 1536,-2048,3072,4096
```

6.2 Creating and Editing Key Categories

Kleopatra allows the user to configure the visual appearance of keys based on a concept called **Key Categories**. **Key Categories** are also used to filter the list of certificates. This section describes how you can edit the available categories and add new ones.

When trying to find the category a key belongs to, Kleopatra tries to match the key to a sequence of key filters, configured in the `libkleopatrarc`. The first one to match defines the category, based on a concept of *specificity*, explained further below.

Each key filter is defined in a config group named `Key Filter #n`, where *n* is a number, starting from `0`.

The only mandatory keys in a `Key Filter #n` group are `Name`, containing the name of the category as displayed in the config dialog, and `id`, which is used as a reference for the filter in other configuration sections (such as `View #n`).

Table 6.1 lists all keys that define the display properties of keys belonging to that category (i.e. those keys that can be adjusted in the config dialog), whereas Table 6.2 lists all keys that define the criteria the filter matches keys against.

Config Key	Type	Description
`background-color`	color	The background color to use. If missing, defaults to whichever background color is defined globally for list views.
`foreground-color`	color	The foreground color to use. If missing, defaults to whichever foreground color is defined globally for list views.
`font`	font	The custom font to use. The font will be scaled to the size configured for list views, and any font attributes (see below) will be applied.
`font-bold`	boolean	If set to `true` and `font` is not set, uses the default list view font with bold font style added (if available). Ignored if `font` is also present.
`font-italic`	boolean	Analogous to `font-bold`, but for italic font style instead of bold.
`font-strikeout`	boolean	If `true`, draws a centered line over the font. Applied even if `font` is set.
`icon`	text	The name of an icon to show in the first column. Not yet implemented.

Table 6.1: Key-Filter Configuration Keys Defining Display Properties

Config Key	Type	If specified, filter matches when...
`is-revoked`	boolean	the key has been revoked.
`match-context`	context[1]	the context in which this filter matches.
`is-expired`	boolean	the key is expired.
`is-disabled`	boolean	the key has been disabled (marked for not using) by the user. Ignored for S/MIME keys.
`is-root-certificate`	boolean	the key is a root certificate. Ignored for OpenPGP keys.
`can-encrypt`	boolean	the key can be used for encryption.
`can-sign`	boolean	the key can be used for signing.
`can-certify`	boolean	the key can be used for signing (certifying) other keys.
`can-authenticate`	boolean	the key can be used for authentication (e.g. as an TLS client certificate).
`is-qualified`	boolean	the key can be used to make Qualified Signatures (as defined by the German Digital Signature Law).
`is-cardkey`	boolean	the key material is stored on a smartcard (instead of on the computer).
`has-secret-key`	boolean	the secret key for this key pair is available.
`is-openpgp-key`	boolean	the key is an OpenPGP key (`true`), or an S/MIME key (`false`).
`was-validated`	boolean	the key has been validated.
`prefix-ownertrust`	validity[2]	the key has exactly (*`prefix`* = `is`), has anything but (*`prefix`* = `is-not`), has at least (*`prefix`* = `is-at-least`), or has at most (*`prefix`* = `is-at-most`) the ownertrust given as the value of the config key. If more than one `prefix-ownertrust` keys (with different *`prefix`* values) are present in a single group, the behavior is undefined.

[1] Context is an enumeration with the following allowed values: `appearance`, `filtering` and `any`.
[2] Validity is an (ordered) enumeration with the following allowed values: `unknown`, `undefined`, `never`, `marginal`, `full`,

`prefix-validity`	validity	Analogous to `prefix-ownertrust`, but for key validity instead of ownertrust.

Table 6.2: Key-Filter Configuration Keys Defining Filter Criteria

> **NOTE**
> Some of the more interesting criteria, such as `is-revoked` or `is-expired` will only work on *validated* keys, which is why, by default, only validated keys are checked for revocation and expiration, although you are free to remove these extra checks.

In addition to the config keys listed above, a key filter may also have an `id` and `match-contexts`.

Using the filter's `id`, which defaults to the filter's config group name if not given or empty, you can reference the key filter elsewhere in the configuration, e.g. in Kleopatra's View configurations. The `id` is not interpreted by Kleopatra, so you can use any string you like, as long as it's unique.

The `match-contexts` limits the applicability of the filter. Two contexts are currently defined: The `appearance` context is used when defining coloring and font properties for the views. The `filtering` context is used to selectively include (and exclude) certificate from views. `any` can be used to signify all currently defined contexts, and is the default if `match-contexts` is not given, or otherwise produces no contexts. This ensures that no key filter can end up 'dead', i.e. with no contexts to apply it in.

The format of the entry is a list of tokens, separated by non-word characters. Each of the tokens is optionally prefixed by an exclamation point (`!`), indicating negation. The tokens act in order on an internal list of contexts, which starts out empty. This is best explained by an example: `any !appearance` is the same as `filtering`, and `appearance !appearance` is producing the empty set, as is `!any`. However, the last two will be internally replaced by `any`, since they produce no contexts at all.

In general, criteria not specified (i.e. the config entry is not set) are not checked for. If a criterion is given, it is checked for and must match for the filter as a whole to match, i.e. the criteria are AND'ed together.

Each filter has an implied 'specificity' that is used to rank all matching filters. The more specific filter wins over less specific ones. If two filters have the same specificity, the one that comes first in the config file wins. A filter's specificity is proportional to the number of criteria it contains.

`ultimate`. See the GPG and GpgSM manuals for a detailed explanation.

Example 6.1 Examples of key filters

To check for all expired, but non-revoked root certificates, you would use a key filter defined as follows:

```
[Key Filter #n]
Name=expired, but not revoked
was-validated=true
is-expired=true
is-revoked=false
is-root-certificate=true
; ( specificity 4 )
```

To check for all disabled OpenPGP keys (not yet supported by Kleopatra) with ownertrust of at least 'marginal', you would use:

```
[Key Filter #n]
Name=disabled OpenPGP keys with marginal or better ownertrust
is-openpgp=true
is-disabled=true
is-at-least-ownertrust=marginal
; ( specificity 3 )
```

6.3 Configuring Archivers for Use with Sign/Encrypt Files

Kleopatra allows the administrator (and power-user) to configure the list of archivers that are presented in the Sign/Encrypt Files dialog.

Each archiver is defined in `libkleopatrarc` as a separate `Archive Definition #n` group, with the following mandatory keys:

`extensions`
: A comma-separated list of filename extensions that usually indicate this archive format.

`id`
: A unique ID used to identify this archiver internally. If in doubt, use the name of the command.

`Name` (translated)
: The user-visible name of this archiver, as shown in the corresponding drop-down menu of the Sign/Encrypt Files dialog.

`pack-command`
: The actual command to archive files. You can use any command, as long as no shell is required to execute it. The program file is looked up using the `PATH` environment variable, unless you use an absolute file path. Quoting is supported as if a shell was used:

```
pack-command="/opt/ZIP v2.32/bin/zip" -r -
```

NOTE
Since backslash (\) is an escape character in KDE config files, you need to double them when they appear in path names:

```
pack-command=C:\\Programs\\GNU\\tar\\gtar.exe ...
```

However, for the command itself (as opposed to its arguments), you may just use forward slashes (/) as path separators on all platforms:

```
pack-command=C:/Programs/GNU/tar/gtar.exe ...
```

This is not supported in arguments, as most Windows® programs use the forward slash for options. For example, the following will not work, since `C:/myarchivescript.bat` is an argument to **cmd.exe**, and / is not converted to \ in arguments, only commands:

```
pack-command=cmd.exe C:/myarchivescript.bat
```

This needs, instead, to be written as:

```
pack-command=cmd.exe C:\\myarchivescript.bat
```

6.3.1 Input Filename Passing for `pack-command`

There are three ways to pass filenames to the pack command. For each of these, `pack-command` provides a particular syntax:

1. As command-line arguments.
   Example (tar):

   ```
   pack-command=tar cf -
   ```

   Example (zip):

   ```
   pack-command=zip -r - %f
   ```

   In this case, filenames are passed on the command line, just like you would when using the command prompt. Kleopatra does not use a shell to execute the command. Therefore, this is a safe way of passing filenames, but it might run into command line length restrictions on some platforms. A literal `%f`, if present, is replaced by the names of the files to archive. Otherwise, filenames are appended to the command line. Thus, the zip Example above could equivalently be written like this:

   ```
   pack-command=zip -r -
   ```

2. Via standard-in, separated by newlines: prepend `|`.
   Example (GNU-tar):

   ```
   pack-command=|gtar cf - -T-
   ```

   Example (ZIP):

   ```
   pack-command=|zip -@ -
   ```

   In this case, filenames are passed to the archiver on stdin, one per line. This avoids problems on platforms which place a low limit on the number of command line arguments that are allowed, but fails when filenames, in fact, contain newlines.

> **NOTE**
> Kleopatra currently only supports LF as a newline separator, not CRLF. This might change in future versions, based on user feedback.

3. Via standard-in, separated by NUL-bytes: prepend `0|`.
   Example (GNU-tar):

```
pack-command=0|gtar cf - -T- --null
```

   This is the same as above, except that NUL bytes are used to separate filenames. Since NUL bytes are forbidden in filenames, this is the most robust way of passing filenames, but not all archivers support it.

6.4 Configuring Checksum Programs for Use with Create/Verify Checksums

Kleopatra allows the administrator (and power-user) to configure the list of checksum programs that the user can choose from in the config dialog and that Kleopatra is able to auto-detect when asked to verify a given file's checksum.

> **NOTE**
> To be usable by Kleopatra, output of checksum programs (both the written checksum file, as well as the output on stdout when verifying checksums) needs to be compatible with GNU **md5sum** and **sha1sum**.
> Specifically, the checksum file needs to be line-based with each line having the following format:
>
> ```
> CHECKSUM ' ' ( ' ' | '*' ) FILENAME
> ```
>
> where *CHECKSUM* consists of hex-characters only. If *FILENAME* contains a newline character, the line must instead read:
>
> ```
> \CHECKSUM ' ' ( ' ' | '*' ) ESCAPED-FILENAME
> ```
>
> where *ESCAPED-FILENAME* is the filename with newlines replaced by `\n`s, and backslashes doubled (\↦\\).
> Similarly, the output of `verify-command` must be of the form
>
> ```
> FILENAME ( ': OK' | ': FAILED' )
> ```
>
> separated by newlines. Newlines and other characters are *not* escaped in the output.[a]
>
> [a] Yes, these programs were not written with graphical frontends in mind, and Kleopatra will fail to correctly parse pathological filenames that contain ″: OK″ plus newline in them.

Each checksum program is defined in `libkleopatrarc` as a separate `Checksum Definition #n` group, with the following mandatory keys:

`file-patterns`
: A list of regular expressions that describe which files should be considered checksum files for this checksum program. The syntax is the one used for string lists in KDE config files.

> **NOTE**
> Since regular expressions usually contain backslashes, care must be taken to properly escape them in the config file. The use of a config file editing tool is recommended.

The platform defines whether the patterns are treated case-sensitive or case-insensitive.

`output-file`

The typical output filename for this checksum program (should match one of the `file-patterns`, of course). This is what Kleopatra will use as the output filename when creating checksum files of this type.

`id`

A unique ID used to identify this checksum program internally. If in doubt, use the name of the command.

`Name` (translated)

The user-visible name of this checksum program, as shown in the drop-down menu in Kleopatra's config dialog.

`create-command`

The actual command with which to create checksum files. The syntax, restrictions and argument passing options are the same as described for `pack-command` in Section 6.3.

`verify-command`

Same as `create-command`, but for checksum verification.

Here is a complete example:

```
[Checksum Definition #1]
file-patterns=sha1sum.txt
output-file=sha1sum.txt
id=sha1sum-gnu
Name=sha1sum (GNU)
Name[de]=sha1sum (GNU)
...
create-command=sha1sum -- %f
verify-command=sha1sum -c -- %f
```

Chapter 7

Credits and License

CONTRIBUTORS

- Marc Mutz mutz@kde.org
- David Faure faure@kde.org
- Steffen Hansen hansen@kde.org
- Matthias Kalle Dalheimer kalle@kde.org
- Jesper Pedersen blackie@kde.org
- Daniel Molkentin molkentin@kde.org