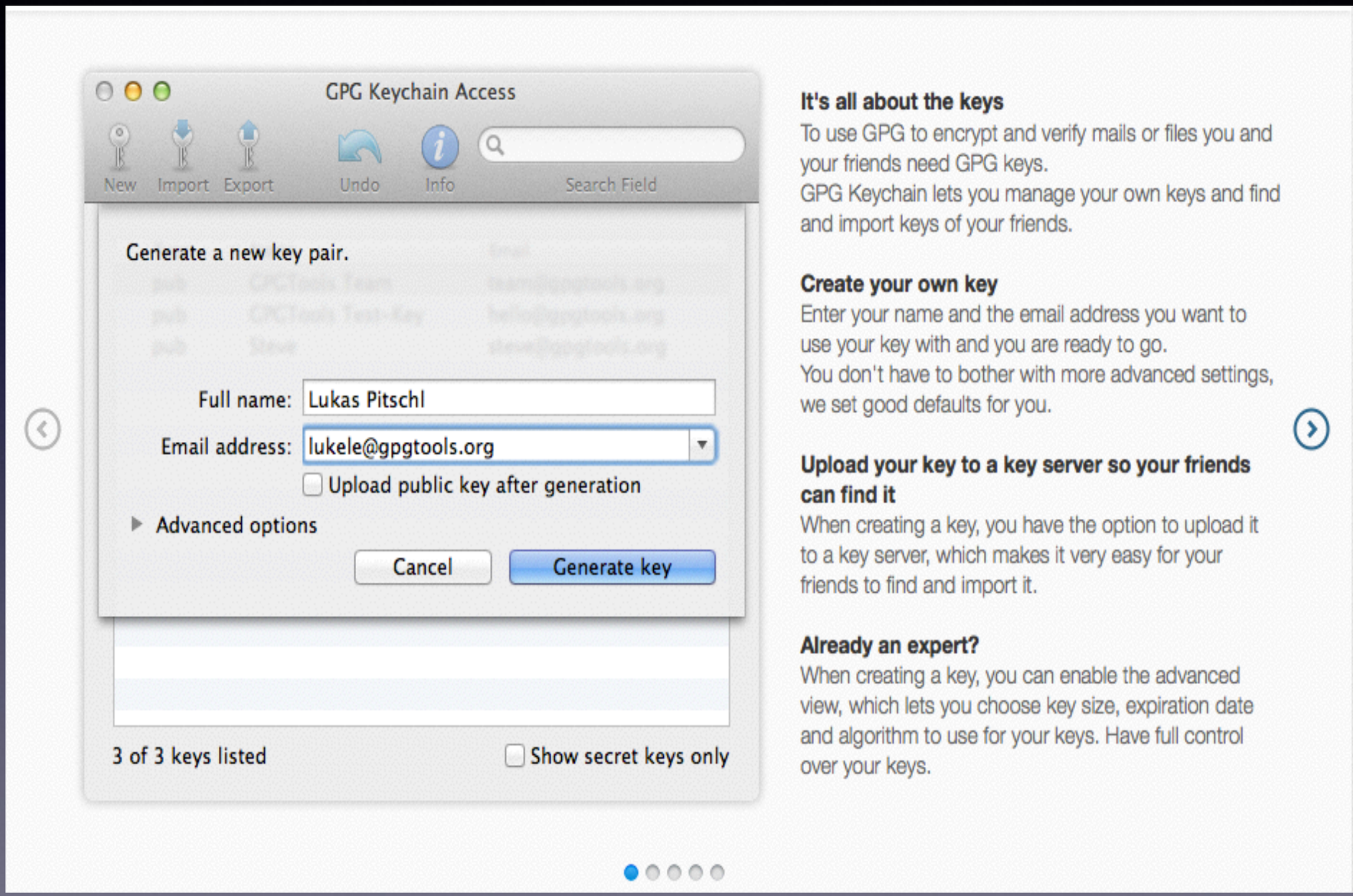
Bypass Pinentry for good via GnuPG, GPGME and Pinentry

Seiya Kawashima
September 9th, 2016

More user friendly message
users might click Cancel button



Source: https://gpgtools.org/

Environment for bypassing pinentry for good

1) GPGME-1.6.0
2) GnuPG-2.0.30, GnuPG modern 2.1.15 as of 09/09/2016
3) Pinentry-0.9.7

Bypassing pinentry for bad ?

```
806 static int
807 dialog_run (pinentry_t pinentry, const char *tty_name, const char *tty_type)
808 {
      /* Comment in all the lines.
         Or replace them with the lines below.
        . . .
       */

      pinentry->pin = (char *)malloc(10);
      memcpy(pinentry->pin,”StrongPIN”,9);
      /* NUL terminate the passphrase.  dialog_run makes sure there is
        enough space for the terminating NUL byte.
        diag.pinentry->pin[diag.pin_len] = 0; */
      pinentry->pin[9] = 0
      pinentry->pin_len = 9;
      return 0;
    }
```

pinentry/pinentry-curse.c

Bypassing pinentry by GnuPG

1) gpg-preset-passphrase command.
2) Flags to cache passphrase in gpg-agent
   such as —max-cache-ttl and —default-cache-ttl

Pros:

1) Good to hide pinentry until explicitly clearing the cache by the users.
2) Good to hide pinentry from the users for a specified period of time.

Cons:

1) Tries to cache as long as years.
2) Needs to repeat specifying the next expiration for the cache.
3) Solves one issue - hiding pinentry.
4) Tightly couple user information and passphrase.

Bypassing pinentry by GnuPG

What users usually would do for the cache

Up to GnuPG 2

7

The user configuration (in `~/.gnupg/gpg-agent.conf`) can only define the default and maximum caching duration; it can't be disabled.

The `default-cache-ttl` option sets the timeout (in seconds) after the last GnuPG activity (so it resets if you use it), the `maximum-cache-ttl` option set the timespan (in seconds) it caches after entering your password. The default value is 7200 (2 hours) for both.

Set it to a year or so – say, 34560000 seconds (400 days) – and you should be fine:

```
default-cache-ttl 34560000
maximum-cache-ttl 34560000
```

But for this change to take effect, you need to end the session by restarting gpg-agent.

If you want to limit to your session length, you'd need to kill the daemon at logout. This is very different between operating systems, so I'm referring to another question/answer containing hints for different systems.

You could also restart the `gpg-agent` during login, but this does not limit caching time to the session length, but logins of a user. Decide yourself if this is a problem in your case.

GnuPG 2.1 and above

In GnuPG 2.1 and above, the `maximum-cache-ttl` option was renamed to `max-cache-ttl` without further changes.

Source: Keep GnuPG credentials cached for entire user session,
http://superuser.com/questions/624343/keep-gnupg-credentials-cached-for-entire-user-session

Bypassing pinentry by GnuPG

1) gpg-preset-passphrase command.
2) Flags to cache passphrase in gpg-agent
   such as —max-cache-ttl and —default-cache-ttl

Pros:

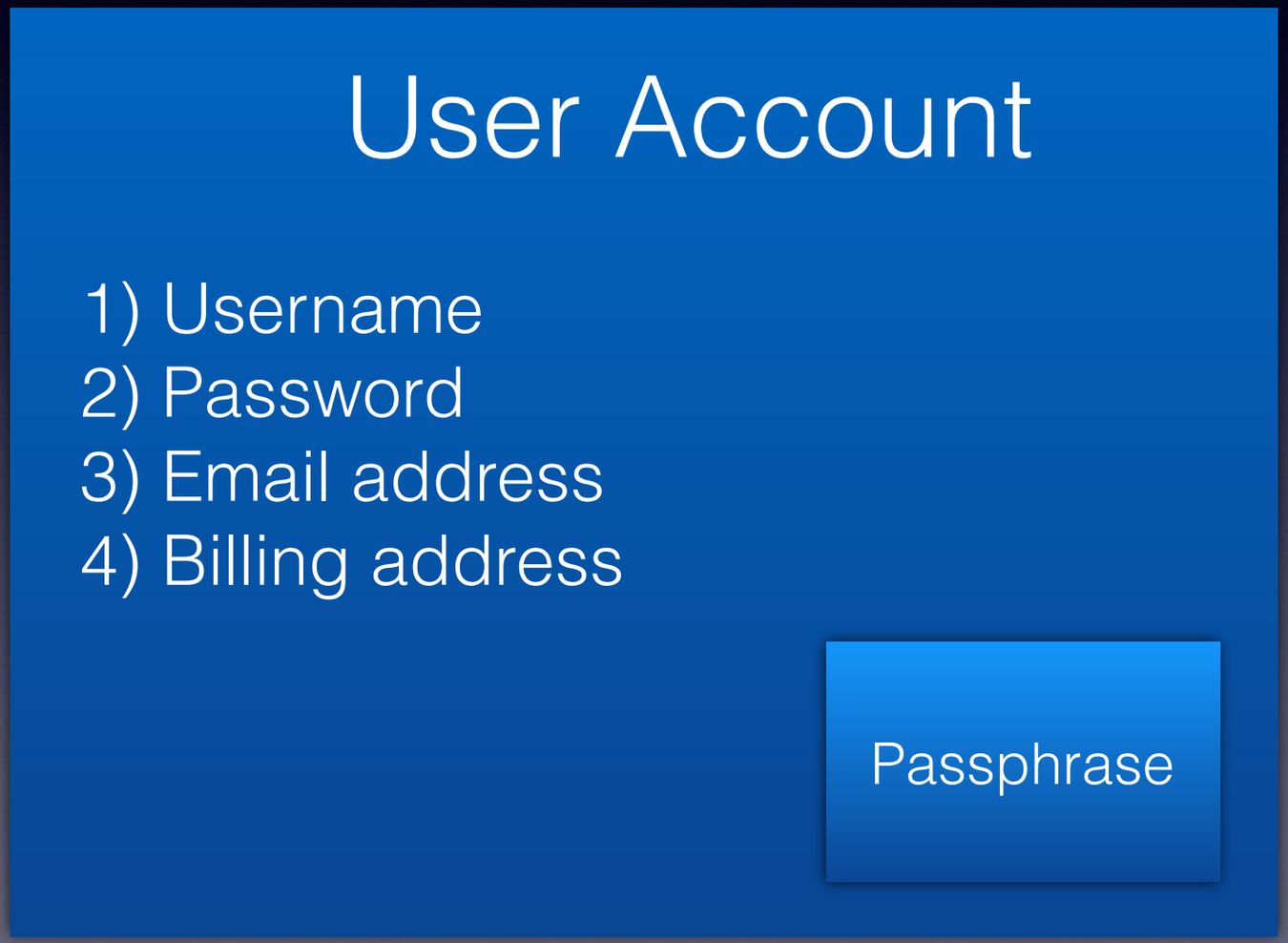
1) Good to hide pinentry until explicitly clearing the cache by the users.
2) Good to hide pinentry from the users for a specified period of time.

Cons:

1) Tries to cache as long as years.
2) Needs to repeat specifying the next expiration for the cache.
3) Solves one issue - hiding pinentry.
4) Tightly couple user information and passphrase.

Bypassing pinentry by GnuPG

Tightly coupling



Bypassing pinentry by GnuPG

1) gpg-preset-passphrase command.
2) Flags to cache passphrase in gpg-agent
   such as —max-cache-ttl and —default-cache-ttl

Pros:

1) Good to hide pinentry until explicitly clearing the cache by the users.
2) Good to hide pinentry from the users for a specified period of time.

Cons:

1) Tries to cache as long as years.
2) Needs to repeat specifying the next expiration for the cache.
3) Solves one issue - hiding pinentry.
4) Tightly couple user information and passphrase.

What GnuPG man says about passphrase

GPG(1) GNU Privacy Guard GPG(1)

NAME

gpg - OpenPGP encryption and signing tool

SYNOPSIS

gpg [**--homedir** dir] [**--options** file] [options] command [args]

DESCRIPTION

gpg is the OpenPGP part of the GNU Privacy Guard (GnuPG). It is a tool to provide digital encryption and signing services using the OpenPGP standard. **gpg** features complete key management and all bells and whistles you can expect from a decent OpenPGP implementation.

This is the standalone version of **gpg**. For desktop use you should consider using **gpg2** ([On some platforms gpg2 is installed under the name **gpg**]).

RETURN VALUE

The program returns 0 if everything was fine, 1 if at least a signature was bad, and other error codes for fatal errors.

WARNINGS

Use a *good* password for your user account and a *good* passphrase to protect your secret key. This passphrase is the weakest part of the whole system. Programs to do dictionary attacks on your secret keyring are very easy to write and so you should protect your "~/.gnupg/" directory very well.

Keep in mind that, if this program is used over a network (telnet), it is *very* easy to spy out your passphrase!

If you are going to verify detached signatures, make sure that the program knows about it; either give both file-names on the command line or use '-' to specify STDIN.

Bypassing pinentry by GnuPG

1) gpg-preset-passphrase command.
2) Flags to cache passphrase in gpg-agent
   such as —max-cache-ttl and —default-cache-ttl

Pros:

1) Good to hide pinentry until explicitly clearing the cache by the users.
2) Good to hide pinentry from the users for a specified period of time.

Cons:

1) Tries to cache as long as years.
2) Needs to repeat specifying the next expiration for the cache.
3) Solves one issue - hiding pinentry.
4) Tightly couple user information and passphrase.

Bypassing pinentry by pinentry-bypass

1) /pinentry-0.9.7/bypass/pinentry-bypass.c

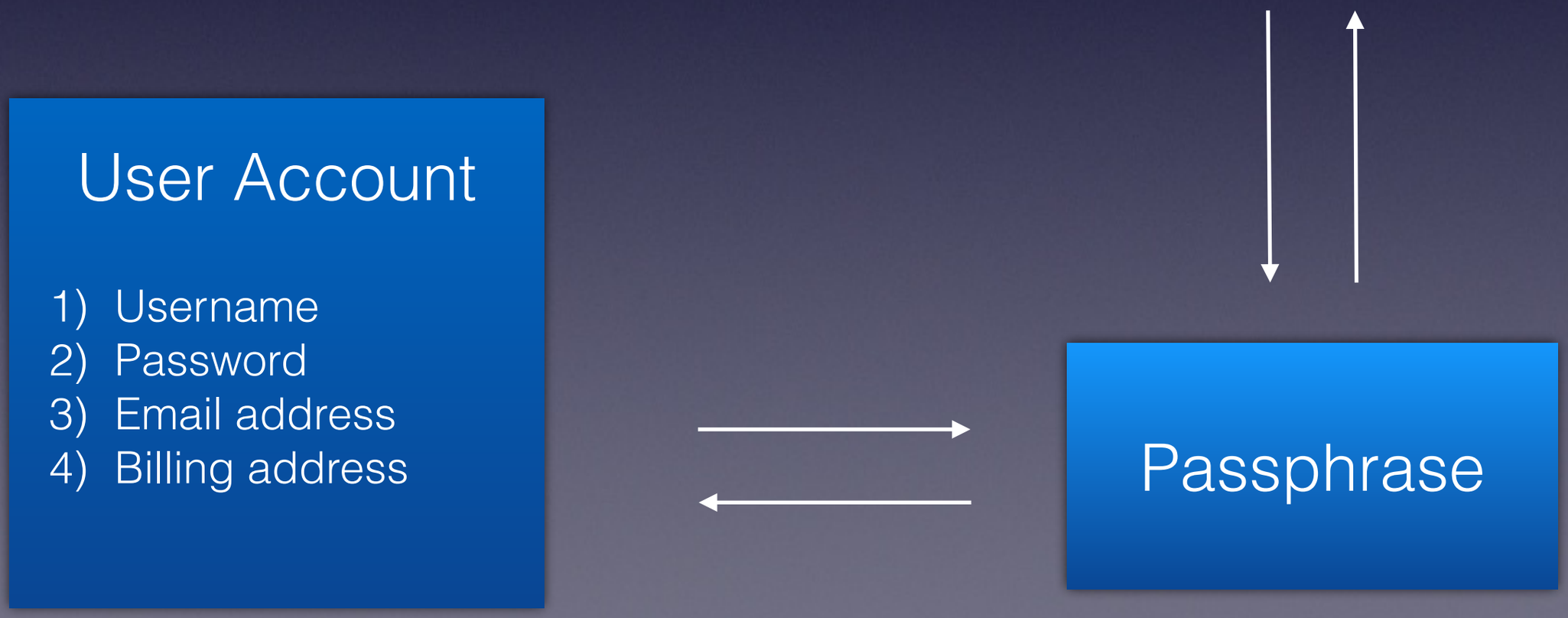
Pros:

1) Good to hide pinentry from the uses.
2) Solves more than one issue.
    1) Hide pinentry from the users.
    2) Loosely couple user information and passphrase.
    3) Generate passphrase for the users.
    4) Regenerate keys for the users.
3) Doesn’t even force users to type and remember their passphrases.
4) Doesn’t require to repeat setting up the next expiration for the cache.

Bypassing pinentry by GnuPG

Loosely coupling

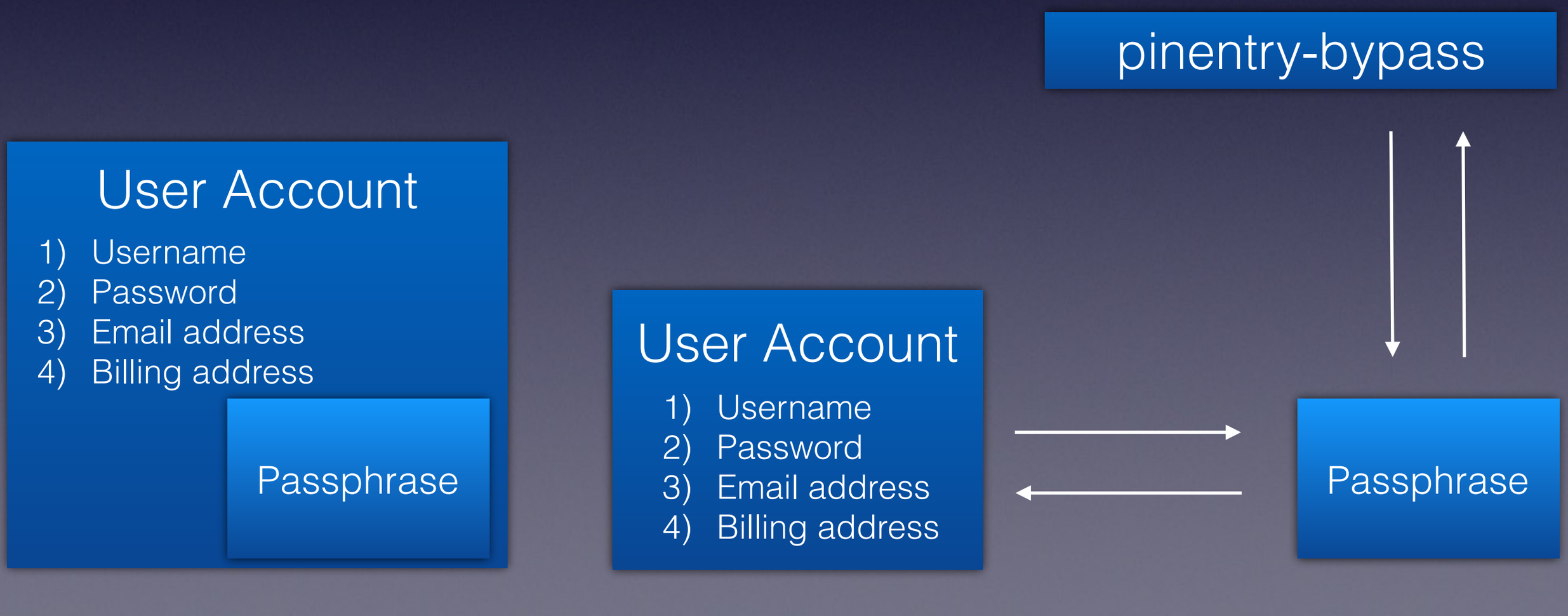
pinentry-bypass



Bypassing pinentry by GnuPG and pinentry-bypass

By GnuPG

By pinentry-bypass



Bypassing pinentry by pinentry-bypass

1) /pinentry-0.9.7/bypass/pinentry-bypass.c

Pros:

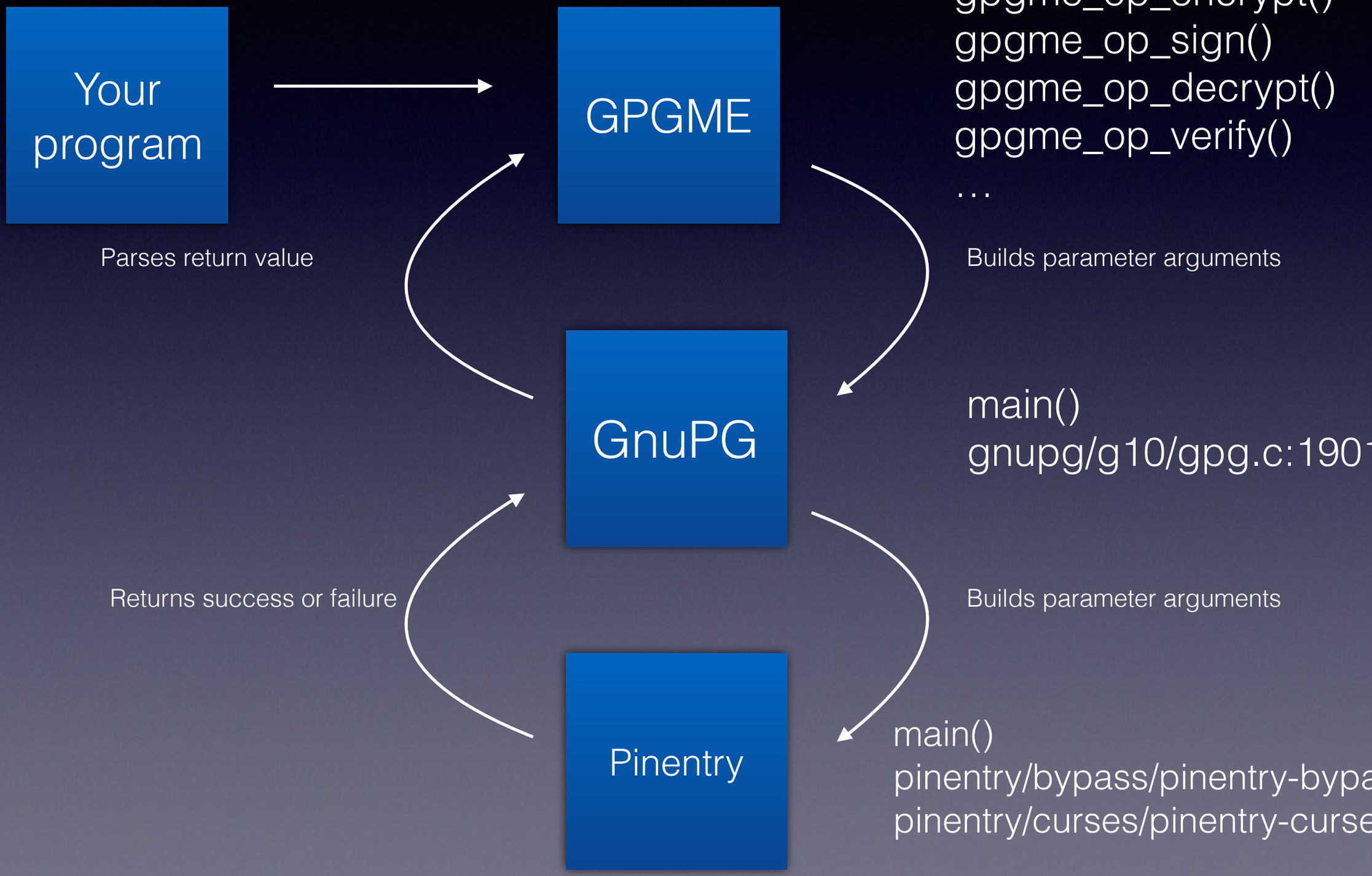
1) Good to hide pinentry from the uses.
2) Solves more than one issue.
    1) Hide pinentry from the users.
    2) Loosely couple user information and passphrase.
    3) Generate passphrase for the users.
    4) Regenerate keys for the users.
3) Doesn’t even force users to type and remember their passphrases.
4) Doesn’t require to repeat setting up the next expiration for the cache.

Cons:

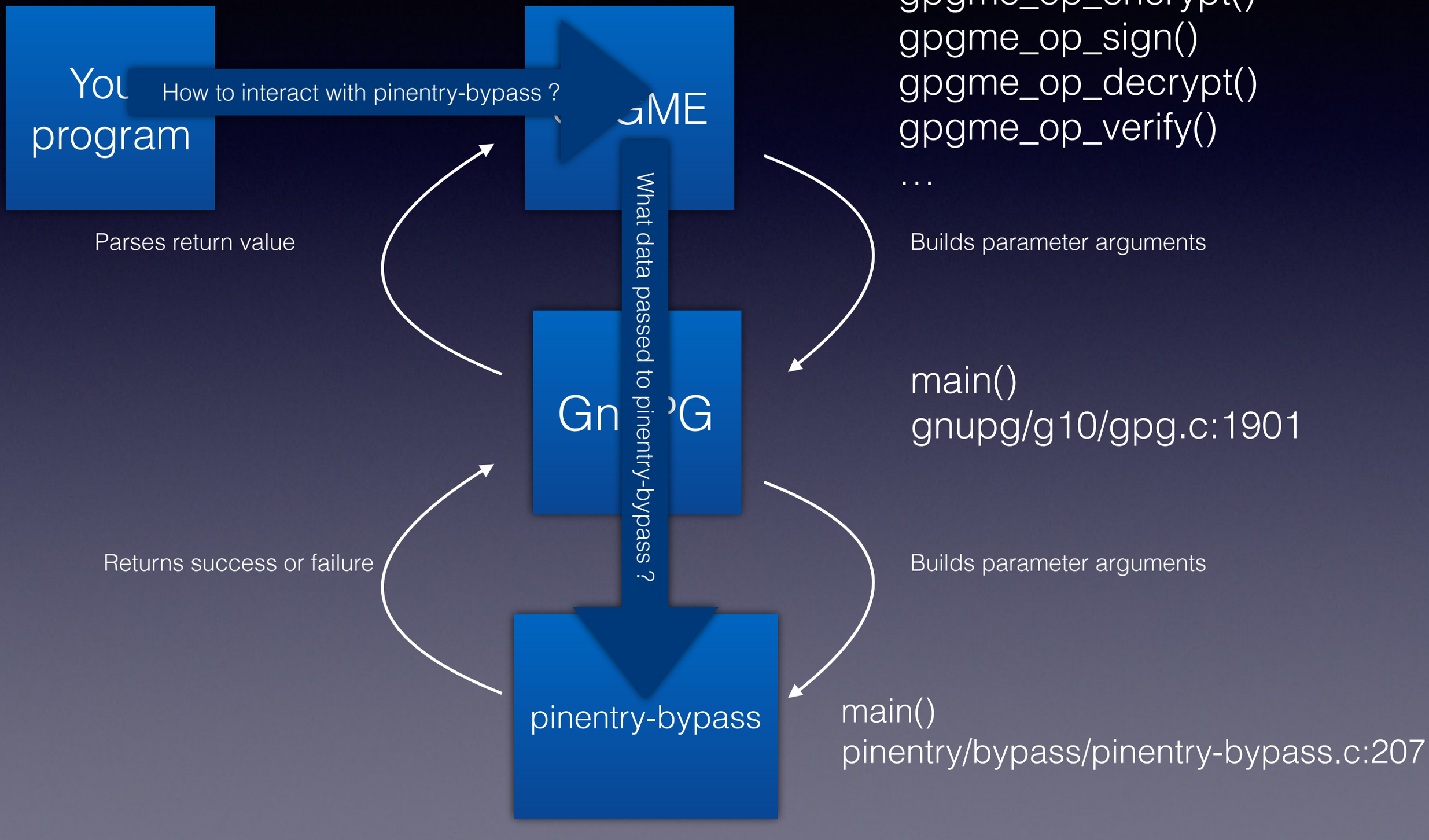
Considered as future work.

1) How to manage generated keys such as Web Key Directory.

Hierarchy of GnuPG's ecosystem



Hierarchy of GnuPG's ecosystem



Overview of what was modified

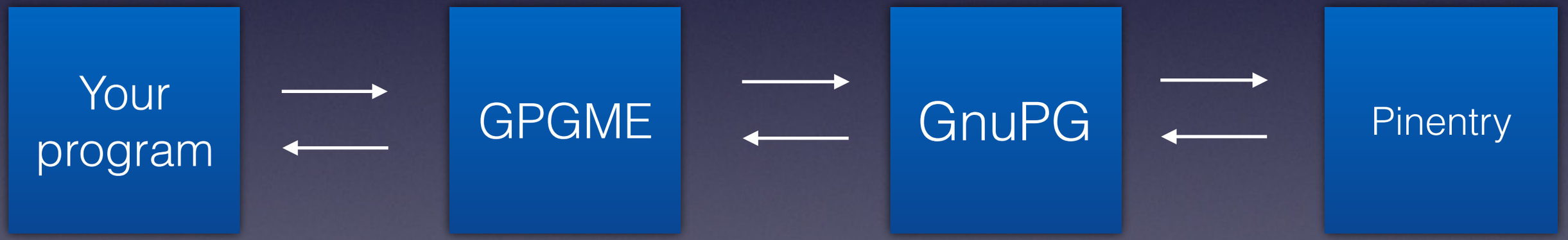
PINENTRY_USER_DATA in gnupg-2.0.30

```
~/gnupg-2.0.30$ grep -nr --include="*.c" "PINENTRY_USER_DATA"
sm/server.c:233:      err = session_env_setenv (opt.session_env, "PINENTRY_USER_DATA",
value);
agent/gpg-agent.c:692:      { "DISPLAY", "TERM", "XAUTHORITY", "PINENTRY_USER_DATA",
NULL };
agent/gpg-agent.c:1361:  session_env_setenv (ctrl->session_env, "PINENTRY_USER_DATA",
NULL);
agent/command.c:1695:    { "GPG_TTY", "DISPLAY", "TERM", "XAUTHORITY",
"PINENTRY_USER_DATA", NULL };
agent/command.c:1963:      err = session_env_setenv (ctrl->session_env,
"PINENTRY_USER_DATA", value);
agent/command-ssh.c:3375:    {"GPG_TTY", "DISPLAY", "TERM", "XAUTHORITY",
"PINENTRY_USER_DATA", NULL};
agent/call-pinentry.c:199:                || !strcmp (name, “PINENTRY_USER_DATA"))
common/simple-pwquery.c:302:  /* Send the PINENTRY_USER_DATA variable.  */
common/simple-pwquery.c:303:  dft_pinentry_user_data = getenv (“PINENTRY_USER_DATA");
common/session-env.c:67:  { "PINENTRY_USER_DATA", "pinentry-user-data"}
```

Overview of what was modified

setenv(
“PINENTRY_USER_DATA”,…)

```
set_opt_session_env
("PINENTRY_USER_DATA",
pargs.r.ret_str)
gnupg/g10/gpg.c:2915
value = session_env_getenv
(ctrl->session_env,
“PINENTRY_USER_DATA")
gnupg/agent/call-pinentry.c:380
asprintf(&optstr,
"OPTION pinentry-user-data=%s”,value)
gnupg/agent/call-pinentry.c:384
```




```
add_arg(gpg,
“-pinentry-user-data")
gpgme/src/engine-gpg.c:580
add_arg(gpg, …)
gpgme/src/engine-gpg.c:582
```

```
ARGPARSE_s_s(
‘u',
"pinentry-user-data",
"|STRING|User data for pinentry”)
pinentry/pinentry/pinentry.c:691
ext_udata(
pinentry->user_data,uds,”;”,1)
pinentry/bypass/pinentry-
bypass.c:132
```

What was modified in GPGME

```
diff --git a/src/engine-gpg.c b/src/engine-gpg.c
index 83befce..9fa9994 100644
--- a/src/engine-gpg.c
+++ b/src/engine-gpg.c
@@ -574,6 +574,15 @@ gpg_new (void **engine, const char
*file_name, const char *home_dir)
       free(tmp);
     }
+  _gpgme_getenv ("PINENTRY_USER_DATA", &tmp);
+  if (tmp)
+    {
+      rc = add_arg (gpg, "--pinentry-user-data");
+      if (!rc)
+        add_arg (gpg, tmp);
+      free(tmp);
+    }
+
   leave:
    if (rc)
      gpg_release (gpg);
```

What was modified in GnuPG

```
diff --git a/g10/gpg.c b/g10/gpg.c
index 97975fb..97d00c8 100644
--- a/g10/gpg.c
+++ b/g10/gpg.c
@@ -375,6 +375,7 @@ enum cmd_and_opt_values
     oAllowMultipleMessages,
     oNoAllowMultipleMessages,
     oAllowWeakDigestAlgos,
+    oPinentryUserData,

     oNoop
   };
@@ -777,6 +778,9 @@ static ARGPARSE_OPTS opts[] = {
   ARGPARSE_s_n (oNoop, "no-sk-comments", "@"),
   ARGPARSE_s_n (oNoop, "no-sig-create-check", "@"),

+  /* User Data passed to pinentry. */
+  ARGPARSE_s_s (oPinentryUserData,   "pinentry-user-data",    "@"),
+
   ARGPARSE_end ()
 };

@@ -2907,7 +2911,9 @@ main (int argc, char **argv)
           case oXauthority:
             set_opt_session_env ("XAUTHORITY", pargs.r.ret_str);
             break;
-
+          case oPinentryUserData:
+            set_opt_session_env ("PINENTRY_USER_DATA", pargs.r.ret_str);
+            break;
           case oLCctype: opt.lc_ctype = pargs.r.ret_str; break;
           case oLCmessages: opt.lc_messages = pargs.r.ret_str; break;
```

What was modified in GnuPG

```
diff --git a/agent/call-pinentry.c b/agent/call-pinentry.c
index 5686998..151a155 100644
--- a/agent/call-pinentry.c
+++ b/agent/call-pinentry.c
@@ -377,6 +377,19 @@ start_pinentry (ctrl_t ctrl)
       if (rc)
         return unlock_pinentry (rc);
     }
+  value = session_env_getenv (ctrl->session_env, "PINENTRY_USER_DATA");
+  if (value)
+    {
+      char *optstr;
+      if (asprintf (&optstr, "OPTION pinentry-user-data=%s", value) < 0
)
+  return unlock_pinentry (out_of_core ());
+      rc = assuan_transact (entry_ctx, optstr, NULL, NULL, NULL, NULL,
NULL,
+            NULL);
+      xfree (optstr);
+      if (rc)
+  return unlock_pinentry (rc);
+    }
+
   if (ctrl->lc_ctype)
     {
       char *optstr;
```

What was modified in Pinentry

1. Set up “—pinentry-user-data” as a parameter argument for pinentry_parse_opts().

2. Added pinentry-0.9.7/bypass/pinentry-bypass.c.

```
typedef int (*pinentry_cmd_handler_t) (pinentry_t pin); /pinentry/pinentry.h:214

/* The caller must define this variable to process assuan commands.  */
extern pinentry_cmd_handler_t pinentry_cmd_handler; /pinentry/pinentry.h:267
```

3. Implemented bypass_cmd_handler().

```
pinentry_cmd_handler_t pinentry_cmd_handler = bypass_cmd_handler; pinentry-bypass.c:203
```

4. Adjusted pinentry-0.9.7/configure.ac.
5. Added pinentry-0.9.7/bypass/Makefile.am

What was modified in Pinentry

```
seiyak@tucana:~/Downloads/pinentry-0.9.7> ./configure -h
`configure' configures pinentry 0.9.7 to adapt to many kinds of systems.

Usage: ./configure [OPTION]... [VAR=VALUE]...

To assign environment variables (e.g., CC, CFLAGS...), specify them as
VAR=VALUE.  See below for descriptions of some of the useful variables.

Defaults for the options are specified in brackets.

Configuration:
  -h, --help              display this help and exit
      --help=short        display options specific to this package
      --help=recursive    display the short help of all the included packages
  -V, --version           display version information and exit
  -q, --quiet, --silent   do not print `checking ...' messages
      --cache-file=FILE   cache test results in FILE [disabled]
  -C, --config-cache      alias for `--cache-file=config.cache'
  -n, --no-create         do not create output files
      --srcdir=DIR        find the sources in DIR [configure dir or `..']

Installation directories:
  --prefix=PREFIX         install architecture-independent files in PREFIX
                          [/usr/local]
  --exec-prefix=EPREFIX   install architecture-dependent files in EPREFIX
                          [PREFIX]
```

Many options come after this ….

What was modified in Pinentry

```
Optional Features:
  --disable-option-checking  ignore unrecognized --enable/--with options
  --disable-FEATURE       do not include FEATURE (same as --enable-FEATURE=no)
  --enable-FEATURE[=ARG]  include FEATURE [ARG=yes]
  --enable-silent-rules   less verbose build output (undo: "make V=1")
  --disable-silent-rules  verbose build output (undo: "make V=0")
  --enable-dependency-tracking
                          do not reject slow dependency extractors
  --disable-dependency-tracking
                          speeds up one-time build
  --enable-maintainer-mode
                          enable make rules and dependencies not useful (and
                          sometimes confusing) to the casual installer
--enable-pinentry-curses
                          build curses pinentry
--enable-fallback-curses
                          include curses fallback
--disable-ncurses       don't prefer -lncurses over -lcurses
--enable-pinentry-tty   build tty pinentry
--disable-rpath         do not hardcode runtime library paths
--enable-pinentry-emacs build emacs pinentry
--enable-inside-emacs   include emacs hack
--enable-pinentry-gtk2  build GTK+-2 pinentry
--enable-pinentry-gnome3
                          build GNOME 3 pinentry
--enable-libsecret      optionally cache passphrases using libsecret
--enable-pinentry-qt    build qt pinentry
--disable-pinentry-qt5  Don't use qt5 even if it is available.
--enable-pinentry-bypass
                          build bypass pinentry
```

What was modified in Pinentry

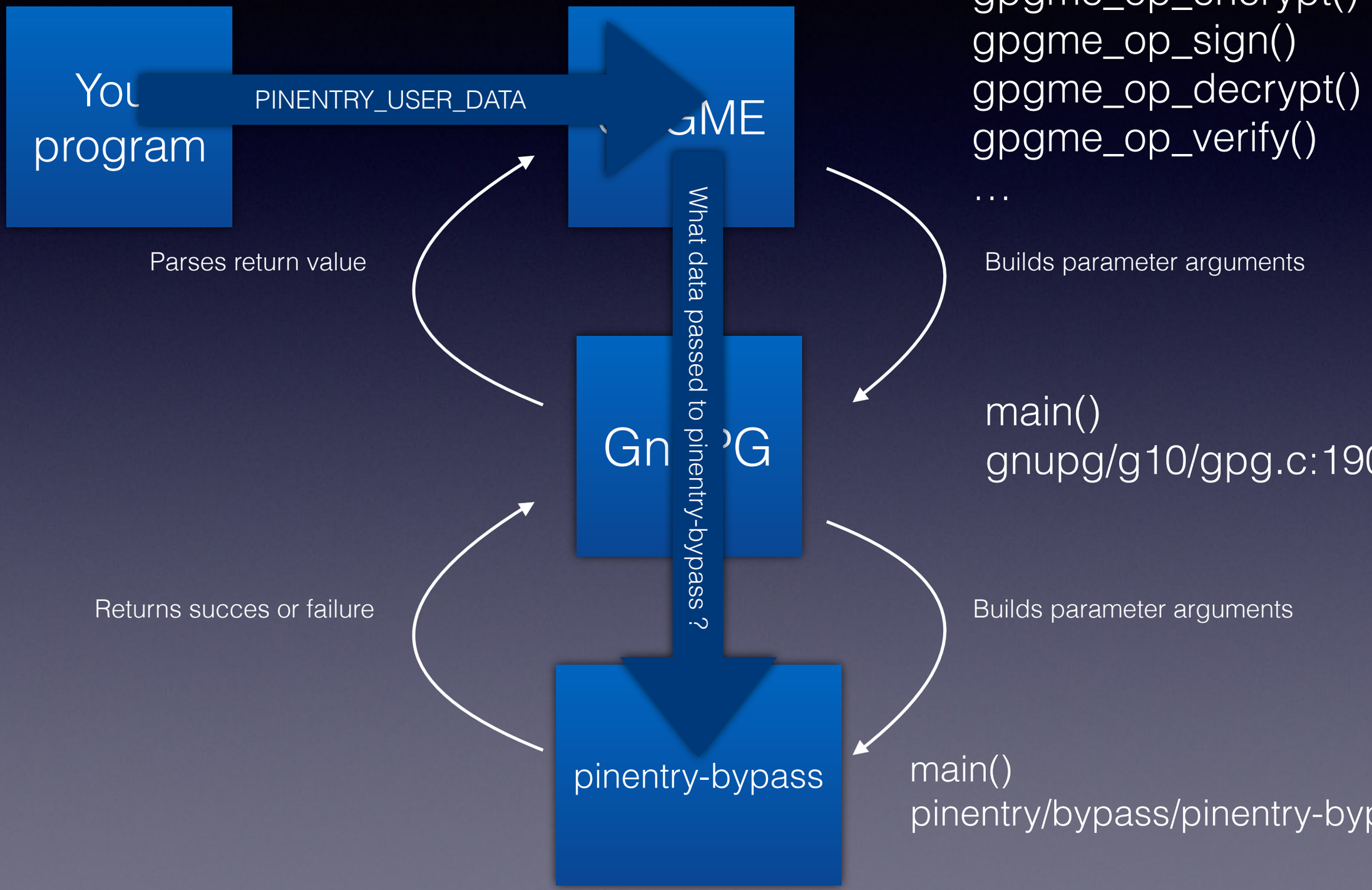
```
seiyak@tucana:~/Downloads/pinentry-0.9.7> ./configure --prefix=/home/seiyak/Downloads/PINENTRY --enable-pinentry-bypass --with-libgpg-error-prefix=/home/seiyak/Downloads
/GPG_ERROR
configure: loading site script /usr/share/site/x86_64-unknown-linux-gnu
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking for style of include used by make... GNU
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking dependency style of gcc... gcc3
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
```

Many outputs come after this ….

What was modified in Pinentry

```
No package 'Qt5Core' found
checking for QtCore >= 4.4.0 QtGui >= 4.4.0... no
checking that generated files are newer than configure... done
checking that generated files are newer than configure... done
configure: creating ./config.status
config.status: creating m4/Makefile
config.status: creating secmem/Makefile
config.status: creating pinentry/Makefile
config.status: creating curses/Makefile
config.status: creating tty/Makefile
config.status: creating emacs/Makefile
config.status: creating gtk+-2/Makefile
config.status: creating gnome3/Makefile
config.status: creating qt/Makefile
config.status: creating w32/Makefile
config.status: creating bypass/Makefile
config.status: creating doc/Makefile
config.status: creating Makefile
config.status: creating config.h
config.status: executing depfiles commands
configure:

        Pinentry v0.9.7 has been configured as follows:

        Revision:    ()
        Platform:  x86_64-unknown-linux-gnu

        Curses Pinentry ..: yes
        TTY Pinentry .....: maybe
        Emacs Pinentry ...: no
        GTK+-2 Pinentry ..: no
        GNOME 3 Pinentry .: no
        Qt Pinentry ......: no
        W32 Pinentry .....: no
        Bypass Pinentry ..: yes

        Fallback to Curses: yes
        Emacs integration : yes

        libsecret ........: no

        Default Pinentry .: pinentry-bypass

seiyak@tucana:~/Downloads/pinentry-0.9.7>
```

Overview of what was modified



Overview of what was modified

PINENTRY_USER_DATA

a;b;c;d;e;f;g;

a) Dynamically linked object file path such as *.so file path.
b) Function name to be called to take care of logic for bypassing pinentry.
c) User defined user role such as 1 for user, 2 for service and 255 for root.
d) User name.
e) GnuPG home directory.
f) Root’s PIN.
g) Root’s name.

Overview of what was modified

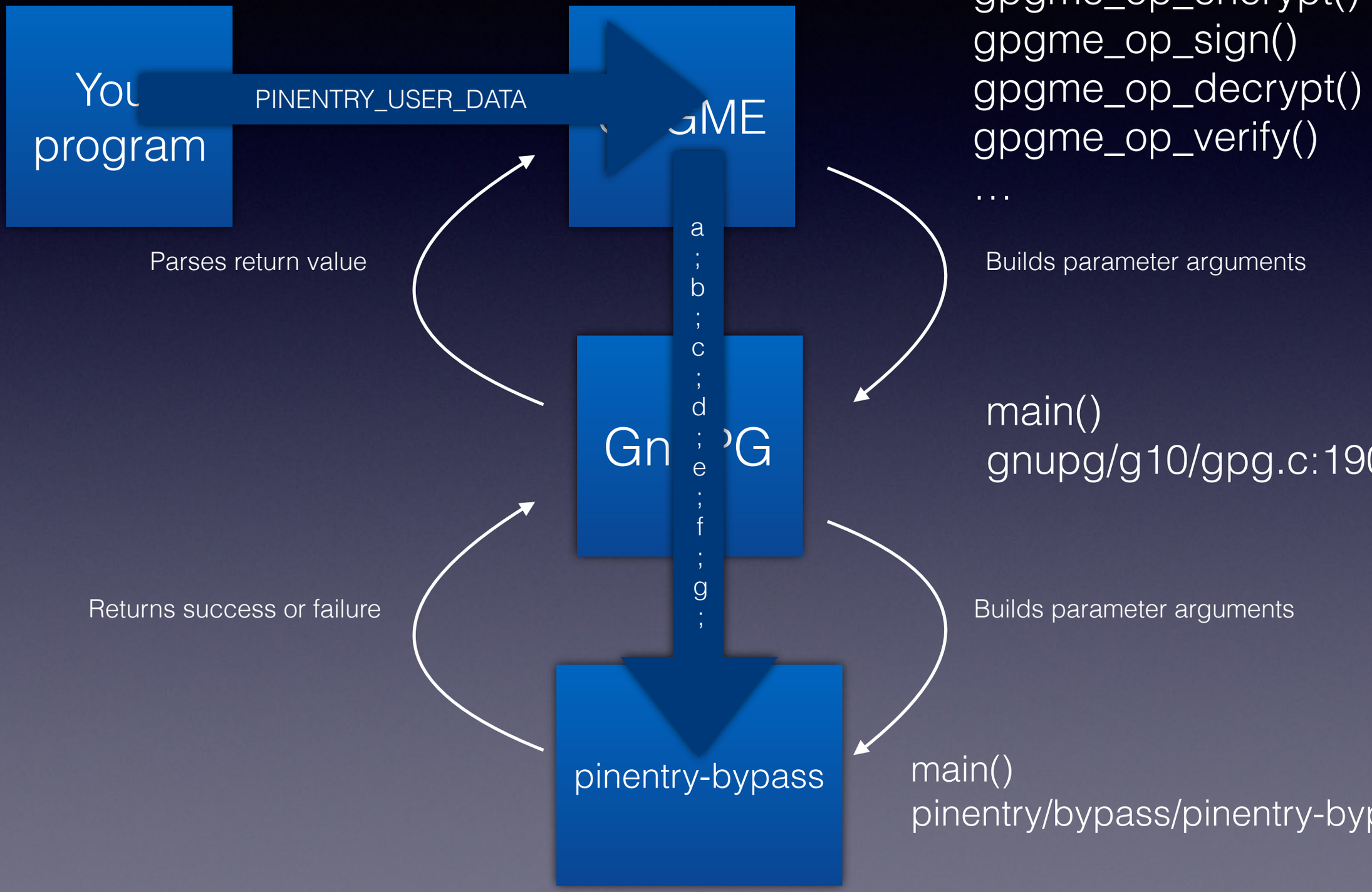
PINENTRY_USER_DATA

File Edit View Bookmarks Settings Help

```
52194 2016-09-06 12:29:51 gpg-agent[27735] starting a new PIN Entry
52195 gpg-agent[27735]: chan_9 <- OK Pleased to meet you, process 27735
52196 2016-09-06 12:29:51 gpg-agent[27735] DBG: connection to PIN entry established
52197 gpg-agent[27735]: chan_9 -> OPTION grab
52198 gpg-agent[27735]: chan_9 <- OK
52199 gpg-agent[27735]: chan_9 -> OPTION ttyname=/dev/pts/16
52200 gpg-agent[27735]: chan_9 <- OK
52201 gpg-agent[27735]: chan_9 -> OPTION ttytype=xterm
52202 gpg-agent[27735]: chan_9 <- OK
52203 gpg-agent[27735]: chan_9 -> OPTION pinentry-user-data=/home/seiyak/Documents/OpenPGP_conf_2016/gpgbyp/resources/lib/liblbyp.so.0.0.1;g
      et_bypass_pin;255;Signer;/home/seiyak/.gnupg;signer;Signer;
52204 gpg-agent[27735]: chan_9 <- OK
52205 gpg-agent[27735]: chan_9 -> OPTION lc-ctype=en_US.UTF-8
52206 gpg-agent[27735]: chan_9 <- OK
52207 gpg-agent[27735]: chan_9 -> OPTION allow-external-password-cache
52208 gpg-agent[27735]: chan_9 <- OK
52209 gpg-agent[27735]: chan_9 -> OPTION default-ok=_OK
52210 gpg-agent[27735]: chan_9 <- OK
52211 gpg-agent[27735]: chan_9 -> OPTION default-cancel=_Cancel
52212 gpg-agent[27735]: chan_9 <- OK
52213 gpg-agent[27735]: chan_9 -> OPTION default-yes=_Yes
52214 gpg-agent[27735]: chan_9 <- ERR 83886254 Unknown option <Pinentry>
52215 gpg-agent[27735]: chan_9 -> OPTION default-no=_No
52216 gpg-agent[27735]: chan_9 <- ERR 83886254 Unknown option <Pinentry>
52217 gpg-agent[27735]: chan_9 -> OPTION default-prompt=PIN:
52218 gpg-agent[27735]: chan_9 <- OK
52219 gpg-agent[27735]: chan_9 -> OPTION default-pwmngr=_Save in password manager
52220 gpg-agent[27735]: chan_9 <- OK
52221 gpg-agent[27735]: chan_9 -> OPTION default-cf-visi=Do you really want to make your passphrase visible on the screen?
52222 gpg-agent[27735]: chan_9 <- ERR 83886254 Unknown option <Pinentry>
52223 gpg-agent[27735]: chan_9 -> OPTION default-tt-visi=Make passphrase visible
52224 gpg-agent[27735]: chan_9 <- ERR 83886254 Unknown option <Pinentry>
```

~/.gnupg/log.log [none,unix] 52216,1 99%52216/522510x0067

Overview of what was modified



Demo

Thank you

Have a great lunch!