PWCRYPT

a Tcl-Program for Keeping Sensitive Data in Encrypted Files

H.-Jürgen Godau

email: `juergen@godau-witten.de`

2006-05-16

Contents

1 Introduction

Several years ago, while working as a programmer at the University of Essen, I had the problems of having to deal with many accounts and login-data and remembering all the passwords and loginnames. I had heard of some *tool* for keeping such data in a file (instead of writing on some paper, which I would never find again in my big pile of paper in my *paperless* office). But there were some problems: it would cost some money, and, even worse, it would only be usable under windows. Since I had done some Tcl-programming before, I decided to write such a program myself and called it *PWCrypt* (an abbreviation for PassWord-enCRYPTion).

2 In the Beginning

First, I had to decide how to store the date so that no *unauthorized* person could read it. I looked for some Tcl-command, but had no luck. By searching the Internet I found some C-program named *enigma*. It was a drop-in replacement for the Unix `crypt` command. It would take a file, prompt for a password and output the (de-)crypted data. The fine thing was, that it used a self-inverse algorithm. So the same command would encrypt data as well as decrypt. Having the C-source, I compiled this program and used the `exec` command of Tcl to (de-)crypt the data. To be able to change the program consistently, I defined some variables at the very beginning (see figure 1). I was quite proud when I

```
# The following two variables ’pwfile’ and ’crypt’ may be
# adjusted according to your operating-system and preferences:

set pwfile "[pwd]/.pwb.dat"
set crypt ./enigma

# Now we define the font used for the listbox and entry-widgets:

option add *Listbox.font 10x20 startupFile
option add *Entry.font 10x20 startupFile

# ********************************************
# * Don’t make any changes beyond this point! *
# ********************************************
```

Figure 1: the prologue (first version)

had the first version ready and saw it was running as well under Linux as under MS-Windows. Since then I regularly used this little helper and was quite satisfied.

3 Evolution

Some time later my bank announced home-banking via Internet. I rushed for getting this enabled because I was tired of having to walk to the bank office for every money-transfer that had to be done. Some time later *eBay* entered the stage and I was very eager to be (more or less useful) things on this rapidly growing platform. But there was a problem: to pay for the beautiful items I *won* at the eBay-auctions, I had to do many money-transfers. And even worse, sometimes I was in my office, sometimes at home. So how to deal with the *TAN*’s (Trans-Action-Number) needed for doing the bank transfers? I would not like to have the list of TANs in my pocket; so I remembered my little password-application and modified it, so that I could keep the TAN-list, received from my bank, in it’s data-file. At that time, the procedure was, that the banking-app asked for *some* TAN from the list, and if this was OK, the transfer would be accepted. So the program had to be added some possibility to *mark* the used TAN’s. It was quite awful to always *change* the information from `<free>` to `<used>`. So I introduced a special TAN-mode, that would *automagically* change the text in the selected entry from `<free>` to `<used>`. Now the next problem arose: When I used the banking at home, I had to take care, not to use the same TAN previously used in the office. So the next step was to add a *reverse* mode, that would show the entries bottom-up instead of top-down! With this I made my money-transfers for some time.

4 into the WEB

As time went by, I got tired of having to transfer the encrypted file between home and office to keep data in sync. So I decided, to make the program into a Web-application. To this purpose I wrote a new version, that would run as a CGI-script under Apache. Using a secured connection (i.E. `https://my.server/ user/pwb.cgi`) I now could see my TAN's from any computer with a browser and an Internet-connection! This was great fun! After some messing with the `tfcrypt`-package I discovered the `TCLLIB` provided i.E. by ActiveTcl. This contained a pure Tcl DES-implementation! After some testing, I was able to use this, so no more external program or library was necessary to run my little app. The WEB-version (called `pwbcgi.tcl`) can also be used as a stand-alone app on the command-line. To decide, weather it is called via Apache or directly, I used the code (inspired by `WiKit`) from figure 2.

```
set pwOpt(cgi) [info exists env(SCRIPT_NAME)]
```

Figure 2: CGI startup-code

The latest version was due round last Christmas. My bank had changed from simple TAN usage to so called iTANs (Indexed Transaction Numbers). This required addition of some code to handle indices in TAN-files. Now you can ask for a specific TAN-number over the NET or on command-line!

5 Interface

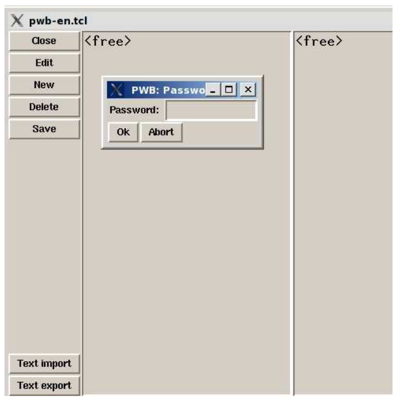
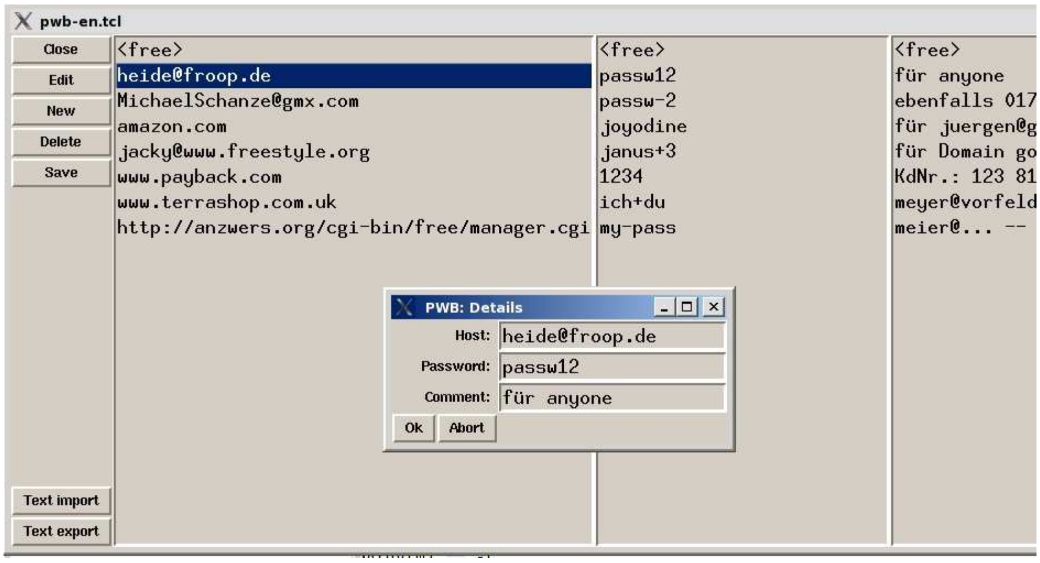


Figure 3: GUI-startup dialogue

After starting the GUI-version you see a display like figure 3. Here you enter the *password* for the used file. When this was correct, you see a display as in figure 4. Here the *change*-button has been hit and the dialogue for editing an entry has popped up. The same dialogue will be used when adding new entries!

Please note the "Import" and "Export"-buttons at the bottom of the main window. This enable you to use simple text-files, created with any text-editor, as basis for encrypted data!

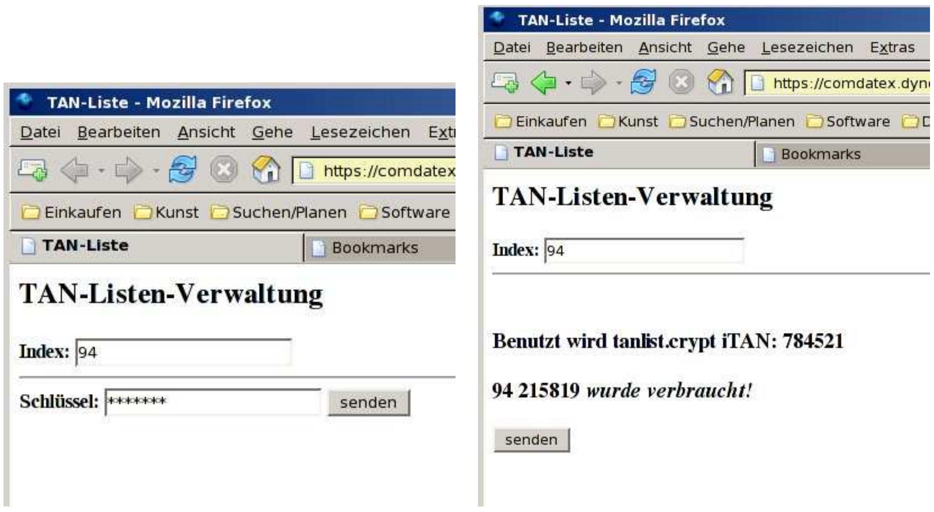
When first started via CGI-call through Apache, *pwbcgi* shows the entry-form from figure 5 (sorry, the labels are in German… Please read "TAN-Listen-Verwaltung" as "TAN-list-management" and "Schlüssel" as "password" resp. "key"). Note that the password is replaced by asterisks! This is achieved by the respective option from ncgi-package when building the form! After transmitting the



<free>	<free>	<free>
heide@froop.de	passw12	für anyone
MichaelSchanze@gmx.com	passw-2	ebenfalls 017
amazon.com	joyodine	für juergen@g
jacky@www.freestyle.org	janus+3	für Domain go
www.payback.com	1234	KdNr.: 123 81
www.terrashop.com.uk	ich+du	meyer@vorfeld
http://anzwers.org/cgi-bin/free/manager.cgi	my-pass	meier@... --

Host: heide@froop.de
Password: passw12
Comment: für anyone

Figure 4: GUI edit dialogue


TAN-Liste - Mozilla Firefox
Datei Bearbeiten Ansicht Gehe Lesezeichen Extr
https://comdatex
Einkaufen Kunst Suchen/Planen Software
TAN-Liste
Bookmarks
senden
TAN-Liste - Mozilla Firefox
Datei Bearbeiten Ansicht Gehe Lesezeichen Extras
https://comdatex.dyn
Einkaufen Kunst Suchen/Planen Software
TAN-Liste
Bookmarks
senden

TAN-Listen-Verwaltung

Index: 94

Schlüssel: *******

TAN-Listen-Verwaltung

Index: 94

Benutzt wird tanlist.crypt iTAN: 784521

94 215819 ***wurde verbraucht!***

Figure 5: web-form

information the password-entry is no longer visible but kept in a hidden field of the form! You can now read one item after the other without re-entering the password. For each desired entry the display shows the right screen from figure 5.

The latest version of `pwbcgi` shows the usage-message from figure 6.

```
 Usage: ./pwbcgi-en.tcl [-t|-i|-u] [-d] [-p passwort] [-f datei]
   -t : TAN-mode
   -i : iTAN-mode (implies ’-t’)
   -u : show next unused TAN & mark as used! (implies ’-t’)
   -r : show entries backwards
   -d : "dump" file
        Default value: ""
resp. "tanlist.crypt" in TAN-mode
```

Figure 6: Usage message of pwbcgi