http://azrar.caracal-cipher.com

Version 1.2.1
February 4, 2014

Manual

http://www.scientific-consilience.com

AZRAR  1.2.1, Manual, , February 4, 2014

Contents

List of Figures

1 Introduction

1.1 At a Glance

Most of today's cryptographic standards will be broken with tomorrow's computational power! AZRAR is a user-friendly tool for the encryption/decryption of single files – preferably compressed archives – addressing **long-term data secrecy**. Thus, it provides the possibility to use webspace and/or email for secure data transfer between two communication/transfer partners.[1] Webspace is often free of charge and seems to be the ideal solution for sharing large amounts of data. Email is commonly free of charge and appears to be the best option for sharing moderate amounts of data very fast. In times of cloud computing and distributed subcontractors these services offer the possibility to link all branch offices together in a business environment. However, at least the provider of webspace/email has the possibility to read and capitalize the transferred data without knowledge of the owner[2]. Furthermore, it can store these data for years and use them eventually in the future when increased computational power allows for breaking current encryption methods. Another aspect concerns the recently uncovered spy activities and data access of security agencies, see, e.g., [1, 2]. Not only competing enterprises practice industrial espionage but it is also rumored that the NSA[3] engages in it [3].

Security requirements in practice depend on precise business demands but are commonly very high in free economy. Individuals do not want everyone to know what they share with friends as well. The task to transfer data as private and secure as possible but viable applies also to Scientific Consilience, a rapidly evolving contract research organization with a strong focus on scientific computing. Especially data secrecy after years when more calculation power is available and allows for cracking today's cryptographic standards is a crucial issue. Thus, we decided to develop a tool which fits business requirements for secure data transfer, especially **long-term data secrecy**.

AZRAR has been developed with the following aims:

- Data in encrypted files are secret/private indefinitely.
- AZRAR supports integrity checking and verification of the transferred data.
- AZRAR supports the user in checking the correctness of entered information.
- AZRAR does not change the system configuration. In fact, AZRAR does not have to be installed. It does not store anything on harddisk – except for key generation.
- AZRAR supports several operating systems natively. There are different versions of AZRAR for each operating system. AZRAR does not require a runtime environment.

[1] Network and email transfer is not yet included in the current version of AZRAR itself and must be done manually via, e.g., `sftp` etc.

[2] This is not a legal debate and far from terms of use or licensing rights. We only state that it has the technical facility.

[3] http://www.nsa.gov/

This tool shall be *free* to use for everyone and *must not* contain any hidden adware or spyware, or (hidden or official) advertisement. Everyone is invited to review the source code and to employ, improve, and to validate this tool. Thus, Scientific Consilience determined to release AZRAR under the GNU General Public License, see Section 5.

AZRAR IS FREE SOFTWARE: YOU CAN REDISTRIBUTE IT AND/OR MODIFY IT UNDER THE TERMS OF THE GNU GENERAL PUBLIC LICENSE AS PUBLISHED BY THE FREE SOFTWARE FOUNDATION, EITHER VERSION 3 OF THE LICENSE, OR (AT YOUR OPTION) ANY LATER VERSION.

AZRAR IS DISTRIBUTED IN THE HOPE THAT IT WILL BE USEFUL, BUT WITHOUT ANY WARRANTY; WITHOUT EVEN THE IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. SEE THE GNU GENERAL PUBLIC LICENSE FOR MORE DETAILS.

YOU HAVE RECEIVED A COPY OF THE GNU GENERAL PUBLIC LICENSE ALONG WITH THIS MANUAL OF AZRAR IN SECTION 5. IF THESE PAGES ARE NOT FINDABLE SEE

http://www.gnu.org/licenses.

1.2 A View Into AZRAR

When it comes to data security/privacy and data encryption/decryption almost all computer users think of a tool where they enter a secret key, passwort, or passphrase and with the aid of this key their data will be encrypted/decrypted. AZRAR follows this conception of using secret information for the en-/decryption of data that can be publically distributed but AZRAR's keys do not consist of single words or phrases (which is insecure and can be attacked by a dictionary approach picking sensible words or dates).

AZRAR is only new in terms of – to the best of our knowledge – no other free software package is available that combines the employed well known approaches and proven facts to a tool that this combination becomes practicable. Furthermore, since it is intended for encryption of data that are then shared via webspace – and at least the provider of the storage service has the technical facilities to store these encrypted data for years without knowledge of the data owners – AZRAR tries to keep these data secret as long as possible. Thus, AZRAR differs from other approaches like PGP (Pretty Good Privacy) [4, 5] or AES (Advanced Encryption Standard, block cipher) [6] in terms of not to rely on effective computational deficiencies. Such viable approaches are based on assumptions about the computational power today. In other words, it takes too long to get the information out of your encrypted data without your key at present. Thus, with more powerful processors the bar is raised and usually the key length is increased. Information in your encrypted data will be readable in a few years without your key, if encrypted with such methods – and data thieves can wait. Furthermore, beside today's remaining risk of potentially existing specialized hardware in a laboratory – currently unknown to the public – most of the public key

ciphers [7, 8] protecting encrypted email, secure web pages, etc. are based on the difficulty of factoring integers [9] or the related discrete logarithm problem and will be broken by quantum computers [10, 11] using, e.g., Shor's algorithm [12, 13] in the future.

AZRAR combines a one-time pad [14, 15] with a stream cipher approach using a cryptographic secure hash function [7, 8] with additional administration processes for practical purposes. The only currently known provable secure data en-/decryption is the so called one-time pad, if used correctly. Once encrypted with this approach your information are perfectly secure for any period of time as long as the key is kept secret. The one-time pad is based on a truly random key that is never reused in whole or part. This key has to be at least as long as the data to encrypt (plaintext). In short, this approach is perfectly secure because without the key all information fitting the size of your data[4] – especially other sensible messages – are equiprobable as your true information [16]. This approach is a symmetric key en-/decryption method since the data have to be encrypted with the same key as they must be decrypted again. For example, the open source software Cryptomni[5] implements the pure one-time pad cipher.

Among others, key sharing is a challenge in practice. A key of the one-time pad approach is at least as large as your data and has to be shared on a secure channel ensuring that a potential attacker will never obtain the key. Furthermore, the key should be stored on a read-only medium, since permanently accessible and not unforgeable key files enable attacks – even if the computer is not online [17, 18]. The most secure method for the transfer is probably to leave the key when you are visiting your communication partner – as secure as you leave your data directly with them – which is impracticable due to time and reaction constraints in almost all cases here considered.

AZRAR uses a secret key which is stored on DVD together with a data transfer code (DTC) list printed on paper. Both should be replaced by new ones if the DTC list is fully used or at least after a few months. Sender and receiver share identical copies of these media by post or courier service on different ways[6] or – *preferably* – *whenever they meet*. The DVD contains enough fresh random bytes for several data transfers. However, the convenience of storing administration profiles and data on harddisk like which DVD bytes can be used and which not for the next encryption is a poor idea, since the whole process of data transmission could be vulnerable. Therefore, AZRAR avoids any electronically stored transfer data, see below. In practice there is the further problem of key collision, i.e. both transfer partners encrypt their data unwittingly and independently with the same random bytes at once. Thus, a key DVD contains two key files – namely `A.key` and `B.key` – assigned to each of the partner for their encryption.

Another task for a system like AZRAR is the unforeseeable size of the data to transfer and very related to practice. Usually all data are compressed to a, e.g., zip archive which is then encrypted by AZRAR. Maybe there are not enough so far unused bytes left on the key DVD but it is not possible to share a new key fast enough due to time constraints. Thus, AZRAR uses an additional

[4] Determination of the true data size seems to be impractical using compressed archives which are then encrypted by AZRAR as recommended.

[5] `http://cryptomni.sourceforge.net/`

[6] Can be attackable by, e.g., security agencies, see [19, 20].

(secret) DTC list separately from the key DVD. This list encodes start positions on the key DVD to prevent an attacker from knowing directly which parts of the DVD have been used twice. Since it might be possible to test whether this occurred all random bytes on DVD are transformed by a cryptographic secure hash function. In this spirit AZRAR acts like a stream cipher [7, 8] with the difference of always incorporating random bytes from DVD. Version 1.2.1 of AZRAR uses Skein, see [21, 22], one of the finalists of the NIST cryptographic hash algorithm competition (2007-2012)[7]. Skein is based on the Threefish tweakable block cipher, a modern successor of Blowfish [23]. This way no part of the key DVD is directly used twice since accumulated fresh random bytes prevent identical key bytes when it comes to multiple usage of DVD bytes. All attacks assuming twice employed bytes for encryption are noneffective. This approach also coincides with the advices in [24] for random numbers, see also [25, 26], and allows also for employing DVD bytes more than twice without dramatically decreasing security.

Additionally, each DTC contains a random string which can be regarded as key for the key and is an additional security facet. Even if DVD bytes are used twice, which is unknown to an attacker, the effective key bytes are different due to different initializing strings. This string contains no sensible words or phrases and, thus, is not attackable by a dictionary approach. AZRAR only uses optically distinguishable characters for user convenience. Nevertheless, version 1.2.1 uses a string size of 11 characters together with about 2.2 billion potential start positions on the key file resulting in a theoretical search space $> 10^{30}$ (number of combinations with starting points for a brute force attack) if the key DVD is known. This means that even if attackers obtained the key DVD they have to try about 10^{30} combinations/starting points in their worst case without the DTC list. *This will be tackable in a few years and is only an additional security facet! Long time data secrecy can be assumed only by keeping secret the key DVD.*

Finally, if key DVD and DTC list are kept secret AZRAR's security facets are:

- Due to the well proven facts and techniques we can commonly expect that no surprising security issue will appear concerning the used methodology.
- Information in encrypted data are secret indefinitely. This can be assumed even considering potential computational power in the future, e.g., quantum computers.
- If key DVD bytes are only used once the data are perfectly secret according to the one-time pad property and Skein's security claims [21].
- AZRAR does not store data on harddisk (except for key generation). Any stored communication process data or session data would enable potential attacks. It runs out of the box without installation.

Furthermore, AZRAR features additional conveniences.

- AZRAR supports short hash codes for checking the correctness of the entered DTC.
- AZRAR supports integrity checking / verification of the transferred data.

[7] http://csrc.nist.gov/groups/ST/hash/sha-3/index.html

- AZRAR supports integrity checking / verification of the key DVD.
- AZRAR implements key and DTC list generation. Although well designed, this implementation may be extended and/or replaced by your own approach.
- AZRAR supports short tooltips and is self-explanatory in many aspects.
- Due to wxWidgets – the employed C++ library for cross-platform development – AZRAR can be compiled for several operating systems supporting natively the feel and look of each operating system.

1.3 About Scientific Consilience

Scientific Consilience[8] is a rapidly evolving contract research organization with a strong focus on scientific computing for biotechnological and pharmaceutical industries. We combine the major findings from various scientific disciplines to generate valuable and unique solutions. Our customers profit from high quality of the results, short development times and reduced running expenses. Key services are

- Biomarker Selection
- Modelling Transdermal Drug Uptake
- Molecular Modelling
- Scientific Computing

More general information can be obtained under

http://www.scientific-consilience.com

or upon request

info@scientific-consilience.com.

The typical scenario is shown in Figure 1. Commonly, a customer owns data that should be processed in a certain way and the results have to be transferred back. As a consequence Scientific Consilience has strong security needs. We have a *strong security policy* for, e.g., data handling and many requirements for secure but viable transfer of huge amounts of frequently changing data. Data changing and update is a key challenge since it appears as often as it is not always practicable to send the new information on DVD. Strict time lines prevent data transfer by post or courier service.

Hence, Scientific Consilience developed AZRAR and determined that everyone should be able to transfer their tables, pictures, data bases, etc. this way.

Scientific Consilience mainly employs AZRAR for the following services.

[8] http://www.scientific-consilience.com

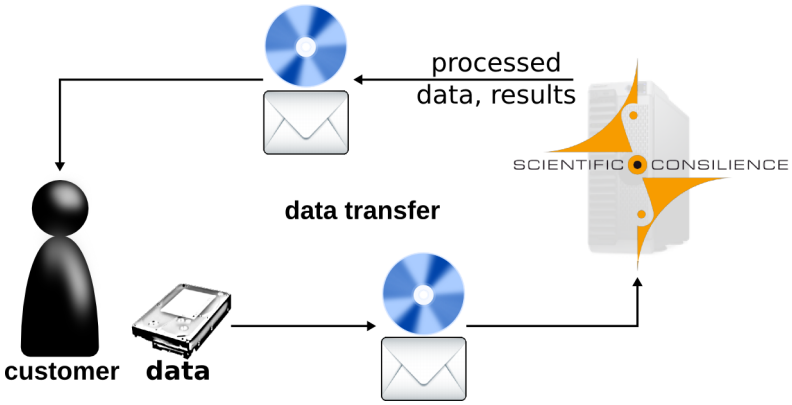

Figure 1: Scientific Consilience – Typical data transfer.

Scientific Consilience: Biomarker Selection

Scientific Consilience offers True**Set**, its proprietary feature subset selection service that encompasses a variety of modern statistical methods to perform feature selection and to construct classification rules.

The reliable identification of variables with predictive power (feature selection) is a key discipline in modern pharmaceutical and biotechnological research. Once these variables have been found, they may be used, e.g., to distinguish cancer tissues from normal tissues, to estimate the aqueous solubility of a substance, or to predict disease states.

In the feature selection process, it is important to avoid irrelevant or redundant features. Including these variables in a predictive model has a detrimental effect on the prediction accuracy, because they may introduce noise without carrying information.

Our software combines information theoretical criteria with statistical evaluations to identify the features with the highest predictive power, see Figure 2. Based on the customer's data, e.g. molecular expression levels, see Figure 3, Scientific Consilience selects relevant biomarkers for the accurate differentiation between several states of diseases. For more information please contact

TrueSet@scientific-consilience.com.

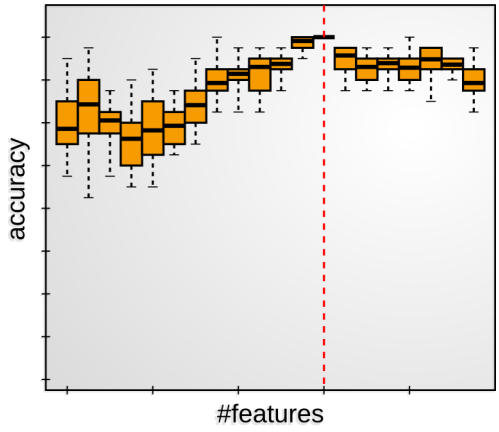

Figure 2: True**Set** – Chosen number of variables with predictive power.

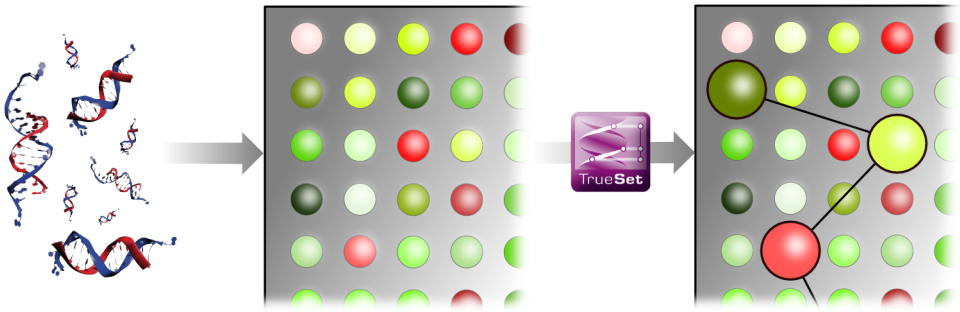

Figure 3: True**Set** – Chosen molecules with predictive power based on microarray data.

Scientific Consilience: Dermal Absorption

A major focus of our services is on modelling and predicting transdermal absorption. Using specialized computational models, Scientific Consilience predicts typical properties of putative drug compounds such as permeability coefficient and lag-time. In addition our approaches allow for estimating concentration-depth profiles in the skin and pharmacokinetics. Based on the results of the calculations we suggest molecular modifications of the putative active pharmaceutical ingredient or of the delivery system to optimize the transdermal absorption.

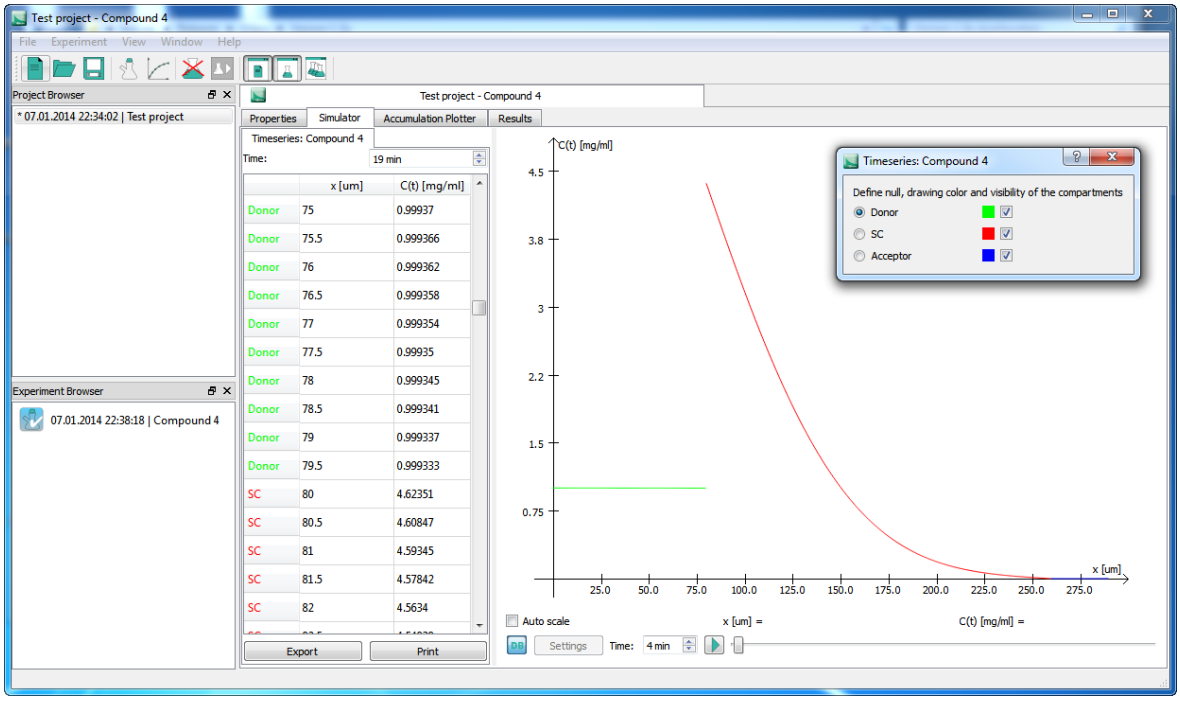

Figure 4: D**Skin** – Computed concentration-depth profile for various times.

We also offer the cross-platform graphical stand-alone software D**Skin** that allows to easily perform Franz-diffusion cell experiments (in vitro test system to investigate transdermal absorption) in silico. Great emphasis was placed on an easy to use graphical user interface (GUI) and high computational speed to allow for running the program on personal computers in a reasonable time. The GUI of D**Skin** allows for easily manipulating the experimental parameters and to examine the results: not only cumulative amounts of substance inside the barrier (typically the stratum corneum) and the acceptor compartment, but also concentration profiles in the stratum corneum and the circumjacent unstirred aqueous layers (see Fig. 4) may be visualized using graphical plots and time-dependent animations. Both plots and raw data can be easily exported. The data for the simulation are predicted internally from the data supplied by the user (diffusant's molecular weight and lipophilicity, system temperature, see Fig. 5). A constant as well as a variable partition coefficient inside the barrier were implemented and can be selected by the user. The underlying diffusion equation was solved numerically using an implicit approach to achieve both high applicational speed and computational precision: For a typical experiment, the calculation time is shorter than one minute.

The model has been extensively validated in a long process using several substances. The software can handle both finite and infinite dose simulations and calculates the permeability coefficient and lag-time for the permeant (infinite dose case). In summary, D**Skin** can be used to simulate penetration as well as permeation experiments with great ease and efficiency due the

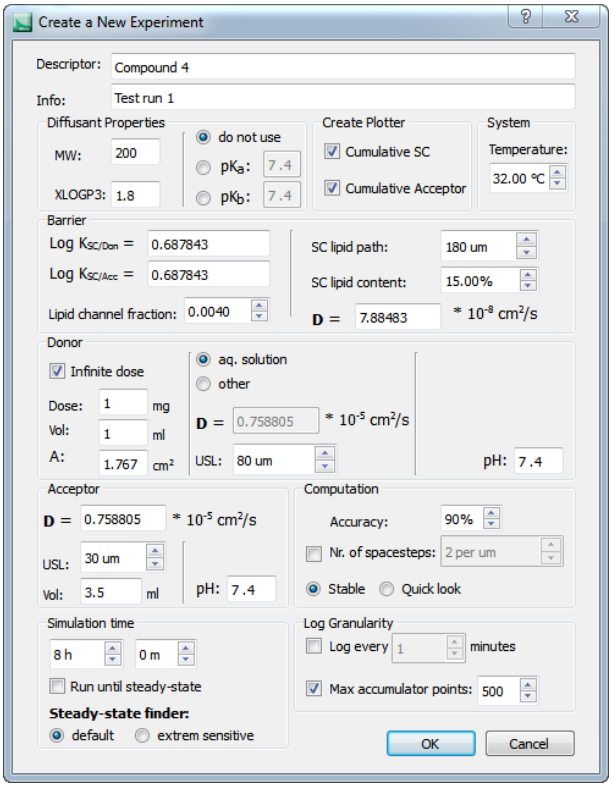

Figure 5: D**Skin** – Settings for virtual experiments.

graphical user interface and the very fast computational speed.

As a special service Scientific Consilience offers to adapt and extend the software to meet the specific needs of customers. Thus, it is possible to use specialized versions of our software for, e.g., the development of transdermal therapeutic systems and for predicting blood plasma levels after application of an ointment after arbitrary times of exposure. For more information, please contact us under

DSkin@scientific-consilience.com.

2 User's Guide

All figures in this manual show AZRAR under Windows XP, if not otherwise stated. AZRAR under other operating systems looks quite similar.

2.1 Installation

2.1.1 Executables

All executables of AZRAR provided by Scientific Consilience should run out of the box without any installation process. Thus, the Windows version is also compatible to Windows UAC. Scientific Consilience compiled wxWidgets[9] and Skein[10] dependencies as well as license or other information into the binaries. On Unix-like systems you should have the GTK+ 2 libraries and dependencies installed, if not otherwise compiled for. Mac OS and Windows come along with their own GUI toolkits and do not need to install any additional library. On many Unix-like systems the Windows version of AZRAR can also be run via WINE[11]. You can download several executables suitable for your operating system at

http://azrar.caracal-cipher.com/downloads.html.

For each download package the hash sums of MD5, SHA1, and SHA-512 are provided. After download please calculate the hash sums and compare your result to the published strings. They must coincide. This is an important step, since it prevents forgery and download errors. Ideally, calculate all three hash sums but at least use SHA-512.

We did not include a self-check in AZRAR (with Skein for example), since this pretends pseudo security. Self-check could be easily forged if an attacker has been able to forge the executable. Furthermore, for a self-check a potentially compromised executable had to be executed. MD5, SHA1, and SHA-512 are hash algorithms most common to the internet.

If you are using an executable compiled by other people please refer to the person who compiled your present binary.

2.1.2 Compiling Source Code

This section gives a few hints for compiling AZRAR from sources. If you have a working executable and do not want to compile AZRAR yourself you may skip this section and further read Section 2.2 on page 13.

For compiling the source code (C++) of AZRAR you must have installed C++ developer tools and libraries for your operating system and have to install wxWidgets, version $\geq$ 2.8. Store the source code of AZRAR in a new folder. The provided source files follow a simple naming

[9] http://www.wxwidgets.org
[10] http://www.skein-hash.info
[11] http://www.winehq.org

scheme: All files beginning with an uppercase letter are AZRAR's own source files, files beginning with a lowercase letter are *Skein's source files and only provided for convenience*. To be exact, the files `brg_endian.h`, `brg_types.h`, `skein_block.c`, `skein.c`, `skein.h`, `skein_iv.h`, and `skein_port.h` are taken from final Skein 1.3 NIST submission (`Optimized_32bit` directory) in October 2010

http://www.skein-hash.info/sites/default/files/NIST_CD_102610.zip.

On Unix-like systems with all developer tools installed you can edit the makefile and customize (if necessary) the line

```
WX_CONFIG ?= wx-config
```

with the path to your wxWidgets config script. Then just typing `make` (GNU make, sometimes also `gmake`, may have to be installed separately, if not available on your system) in your AZRAR directory should build the executable. On Windows this should also work with MinGW and MSYS[12] as well as with Cygwin[13], assuming MinGW/MSYS or Cygwin are properly configured and wxWidgets has been built using these environments (`configure` etc.). On systems without even any Unix-like environment or if you prefer to use an IDE (Integrated Development Environment) create a project using wxWidgets (setting the required dependencies, paths, etc.)[14] add all source and header files to your project (the main files, i.e. the files that implement the application are `AzrarApp.cpp/.h`), and compile your new project. All compilers that are known to be working with wxWidgets should also be working with AZRAR, see

http://wiki.wxwidgets.org/Supported_Platforms.

List of Tested Systems

AZRAR has been successfully tested on several operating systems. Each executable provided by Scientific Consilience should work out of the box on its intended platform assuming the necessary libraries are installed (e.g. GTK+ $\geq$ 2.4 and dependencies if necessary). The number of successfully tested operating systems is increasing. At editorial deadline of this manual (February 4, 2014) the following systems have been reported as successful (verified tests with a much larger number of successful executions in common but not reported to us):

- Windows XP and Vista, all 32 bit
- Windows 7, 64 bit
- Mac OS X 10.7 (Lion), 10.8 (Mountain Lion), 10.9 (Mavericks), all 64 bit
- Oracle Solaris 10 and 11, 32 and 64 bit
- CentOS 5 and 6, 32 and 64 bit
- Debian 7, 32 and 64 bit
- OpenSuse 13.1, 32 and 64 bit

[12] http://www.mingw.org

[13] http://www.cygwin.com

[14] wxWidgets comes with a lot of preconfigured project files for development with Visual Studio (Windows). Code::Blocks (http://www.codeblocks.org), CodeLite (http://www.codelite.org), and other IDEs come along with their own templates for building wxWidgets applications.

- Slackware 14.1, 32 and 64 bit
- Fedora 19, 32 and 64 bit
- Calculate Linux (Gentoo based) 13.12, 32 and 64 bit
- Ubuntu 8.10 and 12.04, 32 bit
- Ubuntu 13.10, 64 bit
- FreeBSD 8.3, 9.2, and 10.0, 32 and 64 bit
- PC-BSD (FreeBSD based) 8.1, 8.2, 32 and 64 bit
- PC-BSD (FreeBSD based) 9.2 and 10.0, 64 bit
- NetBSD 6.0, 64 bit

Known Issues

Assumed you have all necessary libraries installed on your system[15], problems usually only occur by building/installing wxWidgets not AZRAR itself. Thus, please also have a look at

http://wiki.wxwidgets.org/Install.

General On all our building processes with wxWidgets, version 3.0.0, using the wxWidgets ports wxX11/Univ and wxMotif showed poor results, i.e. widgets were deformed up to AZRAR is not usable due to problems with addressing the key directory on the key DVD. The main ports wxMSW (Windows), wxGTK (GTK+ version $\geq$ 2.4 should be available/installable on all Unix-like systems), and wxMac (Mac OS) worked fine. Please note that wxWidgets, version 2.8, might cause errors with recent compilers. If so, please use version $\geq$ 3.0 instead.

Windows On Windows some versions of MinGW[16] have to be patched, i.e. in file `include\commctrl.h` of the compiler directory line 2217

```
#define TV_DISPINFO __AW(NMTVDISPINFO)
```

has to be replaced by

```
#define TV_DISPINFO __AW(TV_DISPINFO)
```

Mac OS X If you are using wxWidgets release version 3.0.0 you may be faced with a bug in the eventloop of wxMac (Cocoa) port. You might have to patch the file `.../src/osx/cocoa/evtloop.mm`, see http://trac.wxwidgets.org/changeset/75378.

Solaris On Oracle Solaris the GNU make tool and dependencies usually have to be installed as well as a compiler, which can be, e.g., the Sun compiler (part of the Oracle Solaris Studio) or the GNU C++ compiler. Make sure you use `gmake` to build wxWidgets by typing `export MAKE=gmake;` (bash) before configuring wxWidgets. Solaris 10 comes with the Java Desktop System (GTK+

[15] For example, GTK+, version $\geq$ 2.4.
[16] For example, MinGW version 4.8.1 and others.

based) and Solaris 11 comes with the GNOME Desktop, so please configure wxWidgets with `--with-gtk`.
Please note that you have at least to count on further installing dependent libraries, if you want to run binaries compiled on versions earlier than 11 on Solaris 11 and vice versa. Oracle Solaris 11 introduced not only a new and more convenient packaging system but also changed dependencies.

Sun CC compiler (Oracle Solaris Studio 12.3). Using the Sun compiler there are no known issues.

GNU C++ compiler. Having only installed the GNU C++ compiler Solaris does not have the file `endian.h`. Thus, the file `brg_endian.h`[17], line 40, has to be patched to include `<sys/isa_defs.h>` instead. For example, replace

```
#if defined( __FreeBSD__ ) || defined( __OpenBSD__ ) || defined( __NetBSD__ )
```

by

```
#if defined(__sun) && defined(__SVR4)
#  include <sys/isa_defs.h>
#elif defined( __FreeBSD__ ) || defined( __OpenBSD__ ) || defined( __NetBSD__ )
```

2.2 Preparation

The system of AZRAR consists of the software itself, a key DVD, and a data transfer code (DTC) list. Both transfer partners share identical copies of the key DVD and the DTC list. The overall process is shown in Figure 6. One of the partners has to realize the following steps:

1. Generate the key and the DTC list.
2. Burn the generated key folder onto DVD (2 copies).
3. Print out the DTC list (2 copies).
4. Send one copy of the key DVD and one copy of the DTC list with *separated*[18] parcels by post or courier service to the communication partner or – *preferably – bring them to her/him personally*.

Both partners store then key DVD and DTC list in a secure place, for example in a safe. Note that keeping the DVD in a dry and dark place at room temperature increases its life time and reduces read errors, which we have to avoid in every case.

Be aware of the fact that if the key DVD is lost or unreadable there is no chance to decrypt corresponding data anymore!

[17] Copyright (c) 2003 Dr. Brian Gladman. This file is used by Skein and only provided for convenience.
[18] This is a security issue. It is less likely that an attacker obtains both, the key DVD and the DTC list.

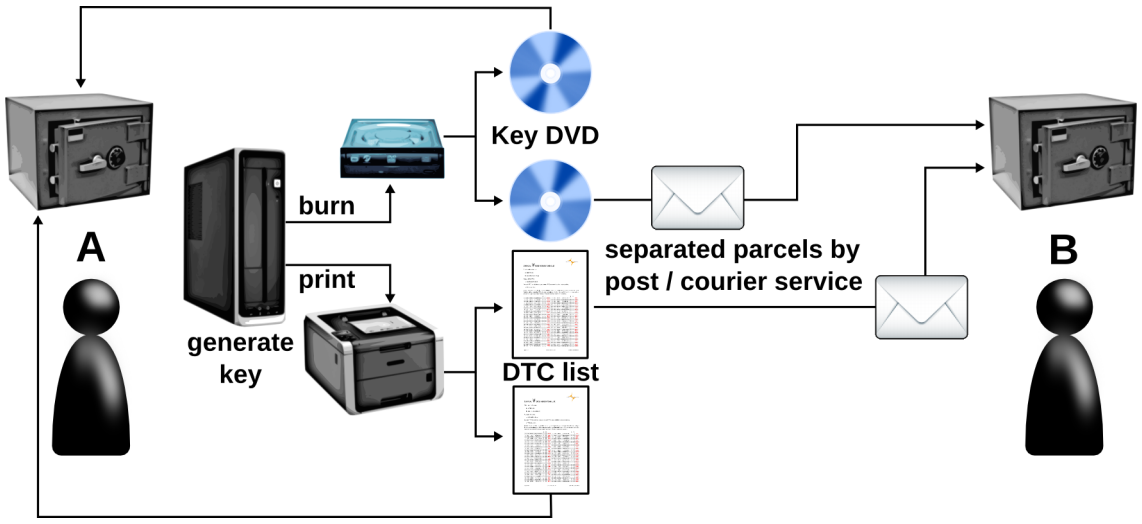

Figure 6: Key generation and key sharing process.

2.2.1 Key Generation

Key generation can easily be done in AZRAR via the `Key` menu entry.

You have to choose (or create) an *empty* folder where your new key will be stored (before burning on DVD and printing the DTC list), the key size in bytes, and the number of DTCs, see Figure 7. The default key size is 2,340,421,632 bytes ($\approx$ 2.2 GB) which is exactly the size to burn all files on a 4.7 GB DVD. You will need to store 2 key files and their hash codes ($\approx 2 \times 2.2$ GB + $2 \times$ 47 bytes). The number of 100 DTCs is default, i.e. for every file to be encrypted we have $\approx$ 2.2 GB/100 = 22 MB *fresh* key bytes for the encryption. More fresh key bytes means more security but less files can be encrypted with one key DVD before being replaced by a new key DVD.

Before the key can be generated the user has to enter a random string. This string is used as additional entropy and is one source, unpredictable for an attacker. Please feel free to play a *jumping monkey* on your keyboard. The longer the entered string the more secure the key.

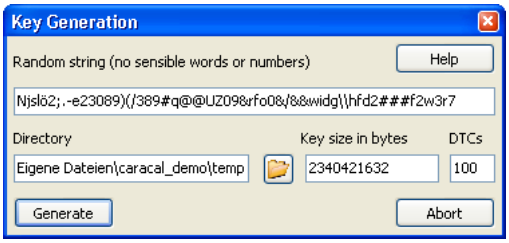

Figure 7: Key generation with AZRAR.

2.2.2 Key Sharing

After key generation you will find a new directory named after the key ID containing four files – `A.key`, `B.key`, `A.hsh`, and `B.hsh` – and the DTC list named as `<key ID>.dtc`. Two different key generation processes usually (but not necessarily) result in two different key IDs. Based on this ID an encrypted file should be assignable to the key DVD even after years. The folder is to be burnt on DVD (the DVD contains this folder regardless of the name of the DVD to be burnt, which lend itself to be the same name/ID). The DTC list is a tabulator separated text file that can be imported in your favorate spreadsheet program to be visually enhanced and printed out on paper. The key files `.key` and their corresponding hash files `.hsh` have to be assigned to the communication partners, see Figure 8.

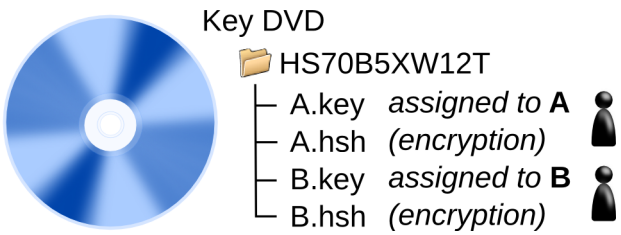

Figure 8: Key file assignment.

All messages or data of partner **A** have to be encrypted with key file `A.key`, all data of **B** with `B.key`, respectively. Both partner use the same DTC list with the specialty that after a DTC is used, the corresponding DTC in the list should be marked off to ensure that every DTC is only used once, see Figure 9. Please also mark off a used DTC of your partner (decryption) to check nonrecurring usage for security reasons. Inform your partner if they used a DTC twice.

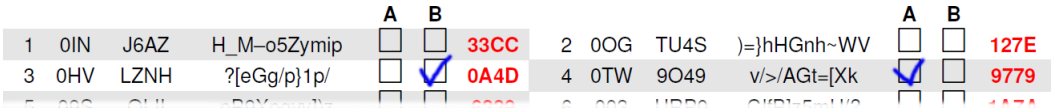

Figure 9: Marking off DTCs.

Note: For security reasons you should never use a key/DTC list for more than one partner you want to share data with. Generate and use one key/DTC list per partner, see Figure 10.

After having burnt the key DVD you and your partner should check the DVD integrity. Insert the DVD in the DVD drive, choose this key directory, go to the tab `DVD Key Integrity` and click on `Generate / Check Hash Code`.

Make sure you selected the directory *on* the DVD, not only the DVD itself! On Unix-like systems it can be confusing to distinguish between the DVD, which can have the same name, and the directory on the DVD, see Figures 11 and 12.

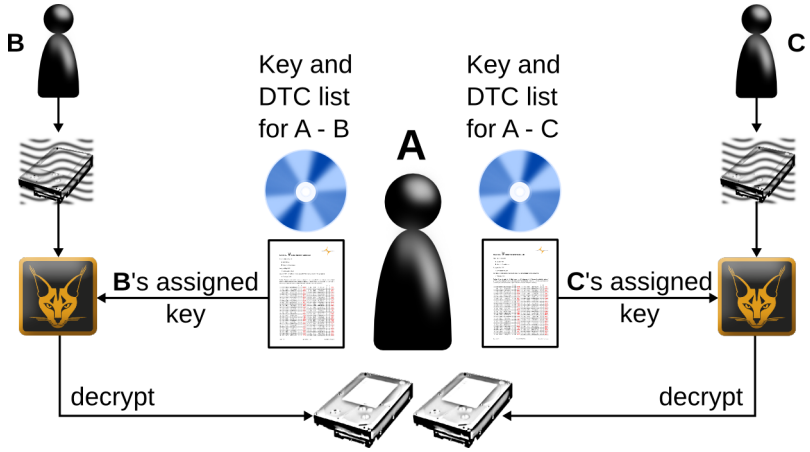

Figure 10: Using AZRAR with two transfer partners.

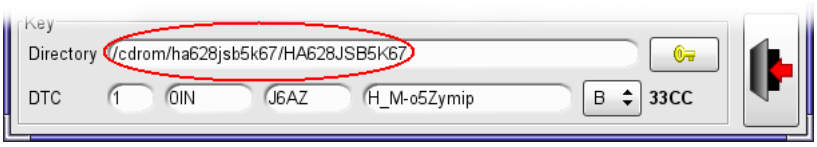

Figure 11: Key directory selected under Solaris 10, Java Desktop System. The key directory is selected on the DVD, which itself is named (mounted as) `ha628jsb5k67` in this example.

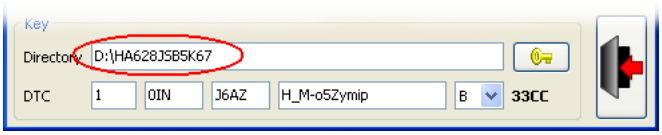

Figure 12: Key directory selected under Windows XP. Windows does not show the name of the DVD itself. Only the directory name is visible.

The reference hash codes are read from DVD (the DVD read sign appears) but can be changed manually (the pressed key sign appears). AZRAR calculates the hash code of the key DVD (files `A.key` and `B.key`), see Figure 13, and compares the results with the entered reference hash codes (if entered). If the hash codes are identical the OK sign appears, see Figure 14, otherwise the red cross stays/appears.

Only use a key DVD if it passed the integrity check!
(Calculated and reference hash codes are identical.)

For security reasons, the default version of AZRAR 1.2.1 warns you before encryption or decryption if the key is not stored on a read-only medium. You should burn your key onto DVD or to

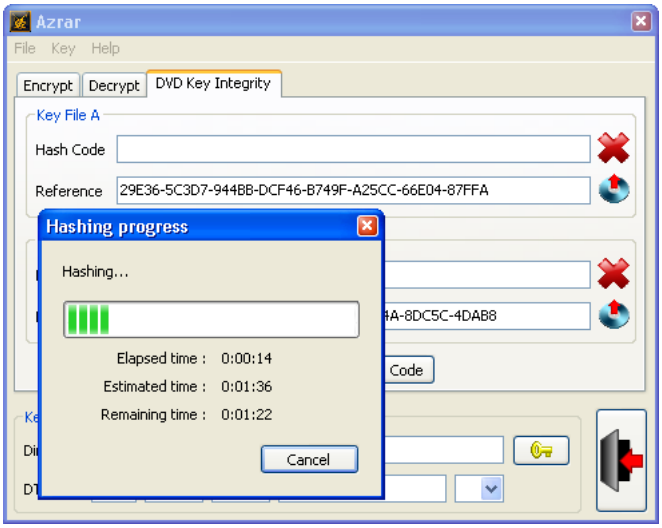

Figure 13: AZRAR calculates the hash codes.

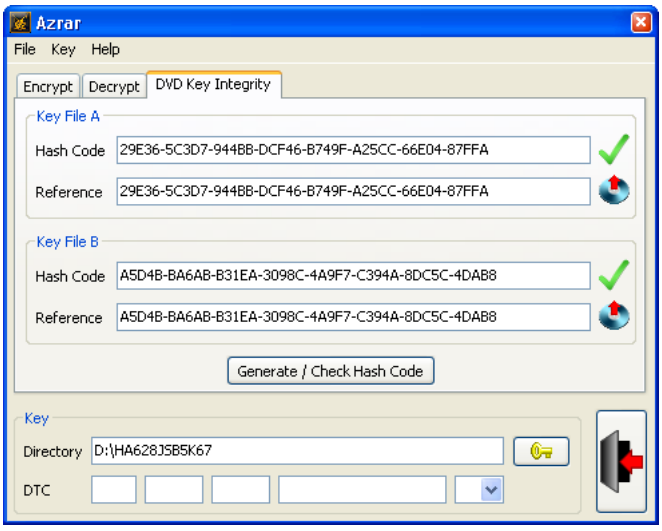

Figure 14: Calculated and compared hash codes. The key DVD is OK.

transfer it onto another read-only medium.

Permanently accessible and not unforgeable key files enable attacks!

Note that it is even technically possible to spy out computers that are not online [17, 18].

Thus, AZRAR remembers you to use the key always from a read-only medium. A warning is printed for each failed read-only check. Future versions of AZRAR may force you to use a read-only medium. *Please insert the key DVD into your drive only for encryption/decryption processes and remove it as soon as possible.* After having checked the integrity of the burnt DVDs you should delete your key from hard drive. For secure file deletion see, e.g.,

`http://sourceforge.net/projects/srm/`

for Unix-like systems or

`http://sourceforge.net/projects/gutmannmethod/`

for Windows.

2.3 Sharing Data

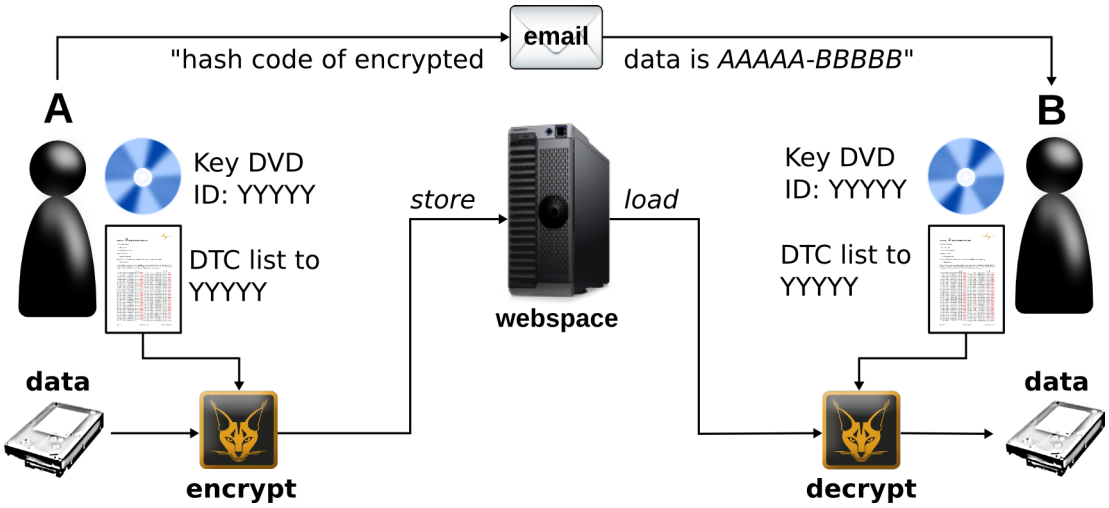

Figure 15: Secure data sharing with AZRAR and webspace.

After having generated and sent key DVD and DTC list, the general process of secure data sharing is shown in Figure 15.

Please compress your data before encryption, i.e. add your files to a compressed archive like, e.g., `zip` or `tar.gz` and encrypt the resulting file. This is a security hint because encrypted compressed data prevent prediction of the size of the original data out of the size of your encrypted data.

2.3.1 Encryption

For the encryption insert the key DVD and choose a not yet employed DTC from the DTC list. At the bottom of AZRAR enter/choose the key directory and choose the character that has been assigned to you, i.e. if you are partner **B** choose B. Enter the DTC and mark off your DTC in the list to prevent multiple usage. *Note:* AZRAR *supports an easy check that the entered DTC is correct: Compare the appeared check code (consisting of 4 letters/digits) on the right with the stated check code right of the DTC in your list. Both codes must be identical.*

Choose the tab `Encrypt`, choose/enter the file to be encrypted (source file, usually a zip archive containing all files you want to share), and enter the filename of the encrypted file. *Note: You do not have to choose a filename that is somehow derived from the original filename. The original filename will be stored encrypted in the encrypted data.* After clicking on `Encrypt / Generate Hash Code` your file will be encrypted, see Figure 16, and the hash code of your encrypted file will be calculated and stated.

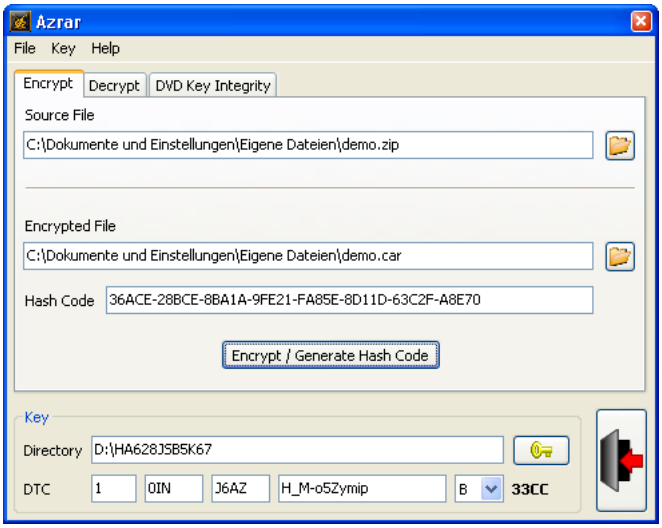

Figure 16: Encryption of a file.

Store your encrypted file on webspace or another location where your transfer partner can reach it. It is sufficient to communicate only the name and the hash code of the encrypted file. The hash code consists of 40 characters (AZRAR version 1.2.1), separated in 8 blocks by "–". It is easily possible to insert the hash code per copy/paste in an email to prevent transcription mistakes.

2.3.2 Decryption

Choose the tab `Decrypt` and choose the encrypted file. Once chosen, AZRAR shows you the DTC number and the key ID as well as the character (A or B) assigned to your partner, which have been used for encryption, above the filename, see Figure 17. Enter the reference hash

code that has been sent by your partner, which can be easily done via copy/paste. Press the check button right from the entered reference hash code. If calculated and reference hash codes are identical the green OK sign appears that shows that we can assume neither transfer error nor forged data.

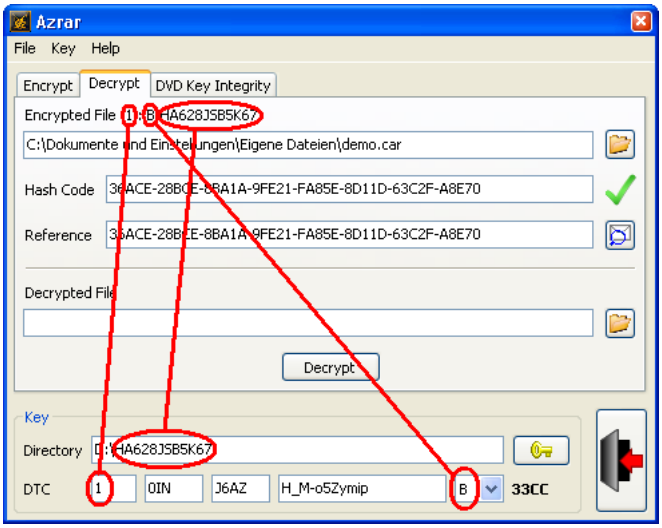

Figure 17: Stored information in encrypted data and where to enter.

Now, insert the key DVD with the given key ID. Enter the character that has been assigned to your communication partner, i.e. if your partner is **B** choose B. This should be the same as stated above the name of the encrypted file. Otherwise your partner might have made a mistake. In every case use the stated character and *if not the same as assigned to your partner communicate the mistake. It might be an attempt of forgery.*

Find the DTC number (first block of DTCs) in the DTC list, enter the DTC and mark off the used DTC in the list in order to determine potential multiple usage. *Please inform your data transfer partner, if you determine multiple usage!* Compare the appeared check code after entering the DTC with the code given in your DTC list to ensure you entered the DTC correctly.

Choose/enter a filename for the decrypted file. Press `Decrypt`, see Figure 18. *Note: Since the filename for the encrypted file is not necessarily linked to the original filename, the original filename is also stored encrypted in the encrypted data.* AZRAR *shows you the original filename right of the DTC number and the key ID during and after the decryption. Maybe you want to rename your stored file to the original filename.*

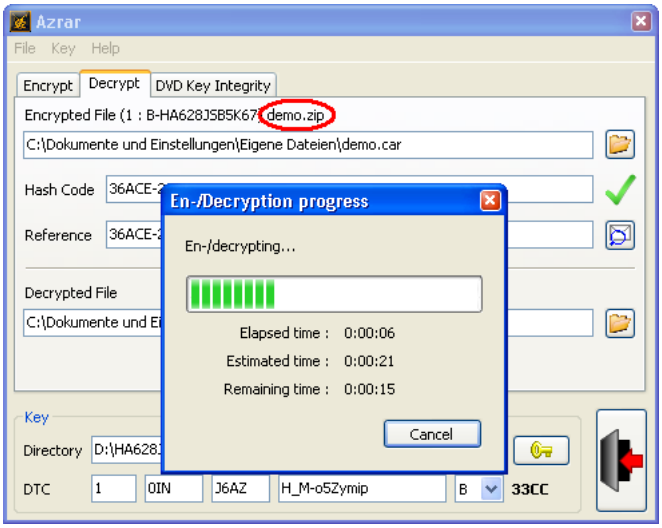

Figure 18: Decryption of a file. The original filename is shown right of the other information.

3 Technical Details

3.1 Key Generation

The key generation is a critical process since cryptographic secure random numbers must be generated.

The key generation process is carefully designed.
However, it may be replaced by your own routines.

Following some advices of operating system developers it would be sufficient to use operating system supplied random numbers, usually based on collected user interaction data or other unpredictable processes. This might be true for some systems while it can take very long time to generate two 2.2 GB key files, if entropy still has to be collected.

On Unix-like systems `/dev/random` provides high quality random numbers but the device blocks under, e.g., Linux until additional environmental noise is gathered. Its counterpart `/dev/urandom` does not block but the random quality depends on the operating system. While `/dev/urandom` is still intended as a pseudorandom number generator suitable for most cryptographic purposes, it is not recommended for the generation of long-term cryptographic keys.

On some Unix-like systems `/dev/random` never blocks and `/dev/urandom` is linked to `/dev/random`. For example FreeBSD and Mac OS X implement the Yarrow algorithm [27]. OpenBSD uses an algorithm based on RC4, called ARC4 providing a fail-safe, i.e. it ensures to provide high quality pseudorandom numbers even on a low entropy pool.

Under Windows the `CryptGenRandom` function of Microsoft's Cryptographic Application Programming Interface[19] can be used.

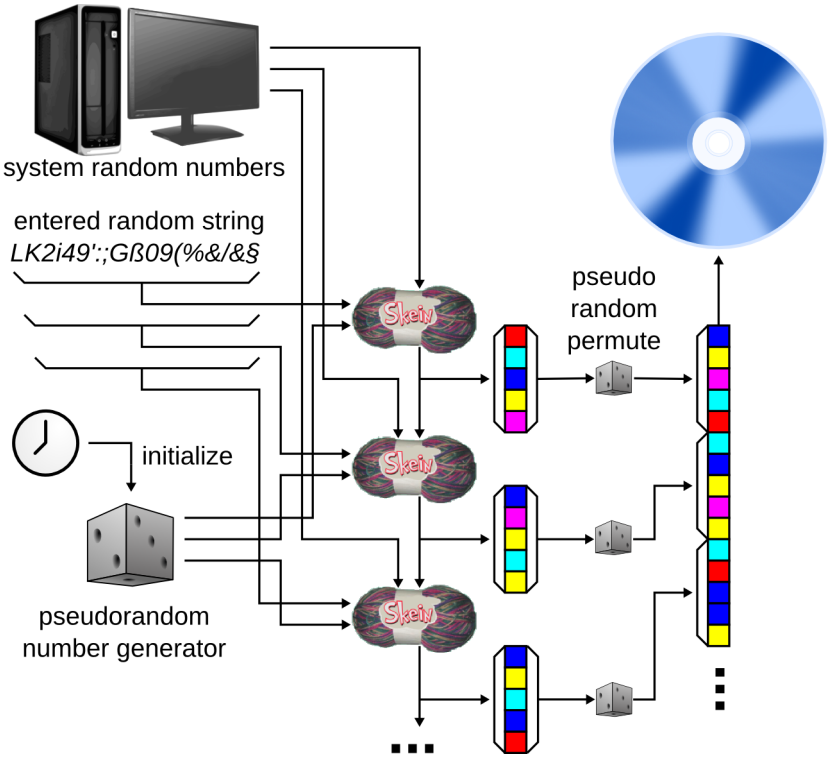

Figure 19: Key generation.

AZRAR's key generation process is the same on all systems. Initial (pseudo)random numbers are transformed via hashing and incorporating additional entropy. On Unix-like systems `/dev/random` is used only a moderate number of times, all other initial numbers are filled by `/dev/urandom`. This approach is valid on systems where `/dev/random` is a blocking device as well as on systems where `/dev/random` and `/dev/urandom` are both non-blocking and in fact the same. However, if `/dev/random` is a blocking device key generation might be speed up by mouse movements or internet browsing of the user. Under Windows the `CryptGenRandom` function initializes the numbers.

Following the advices in [24] and considering the diverse qualities of system supplied random numbers, we use Skein-1024 [21, 22] as pseudorandom number generator after initializing together with the random string entered by the user. Furthermore, an additional pseudorandom number generator is employed for shuffling data and as a kind of counter for Skein. The pseudorandom number generator is seeded by the microseconds of the current time. Its purpose is only data confusion and, thus, it is cryptographically uncritical here due to the quality of all other

[19] http://msdn.microsoft.com/en-us/library/aa380256.aspx

components, so we do not need to use a further cryptographically secure generator. However, it might be replaced by your own pseudorandom number generator. The overall process is shown in Figure 19.

3.2 DTC List

The DTC list consists of the DTCs. A DTC itself contains the DTC number to identify the DTC used for the encryption, the start position on the key file, and an additional random string (usage will be explained in the next Section 3.3), see Figure 20.

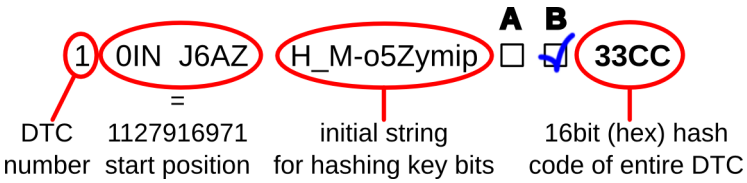

Figure 20: The parts of a DTC.

Start positions are about (not exact) uniformly distributed over the key files. Key files are used as circle, see Figure 21.

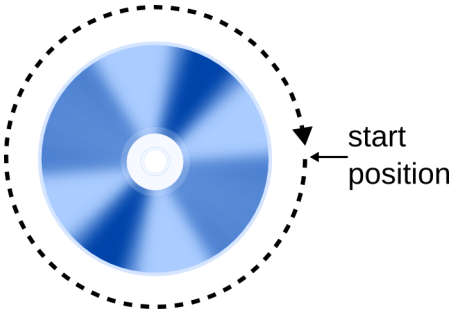

Figure 21: Circular usage of key files.

The start position is encoded as base 36 number, i.e. the digits 0 – 9 and letters A – Z. The DTC list is identical for both communication partners but their assigned character refers to the corresponding key file, see Figure 22.

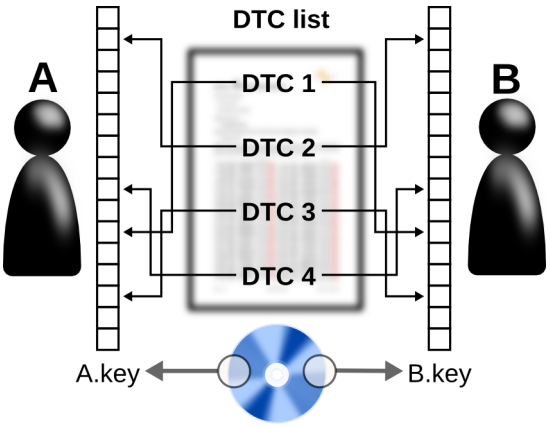

Figure 22: DTC list is identical for both partners.

3.3 Encryption and Decryption

Encryption as well as decryption is done by exclusive or (Xor) of data and key bytes. As explained in Section 1.2 if data size is too large we are using parts of the bytes on the key DVD multiple times. In order to prevent attacks we do not use the DVD bytes directly. The procedure is shown in Figure 23.

The process is a mixture of a stream cipher using Skein-1024 with potentially unused random bits and a one-time pad. The key in terms of a stream cipher is the 4th block of the DTC. The mode of operation is similar to a combination of output feedback encryption and counter mode [7, 8, 28, 29] with the difference of incorporating DVD bytes. The 32 bit unsigned counter is used in little-endian format [30]. We can assume [21] that hashing via Skein using cryptographically secure pseudorandom numbers from DVD yields cryptographically secure pseudorandom numbers again. Thus, if the region on the key DVD is only used once we have a one-time pad. Furthermore, no part of the key DVD is directly used twice since accumulated data prevent identical derived key bytes when it comes to multiple usage of DVD bytes.

3.4 File Format

AZRAR stores in the files of the encrypted data additional information as shown in Figure 24.

3.5 Security Note

The most critical process is key generation. We cannot assume to run on an operating system that provides under every circumstances high quality (unpredictable) random numbers. For example, there are several ways to attack RC4 and derived systems [31, 32]. In an insecure environment we cannot even expect to get numbers that look random. Most probably the best choice to address the problem is to follow the hints in [24], i.e. hashing the pseudorandom numbers with a cryptographic hash function. A good approach is to incorporate additionally as many entropy

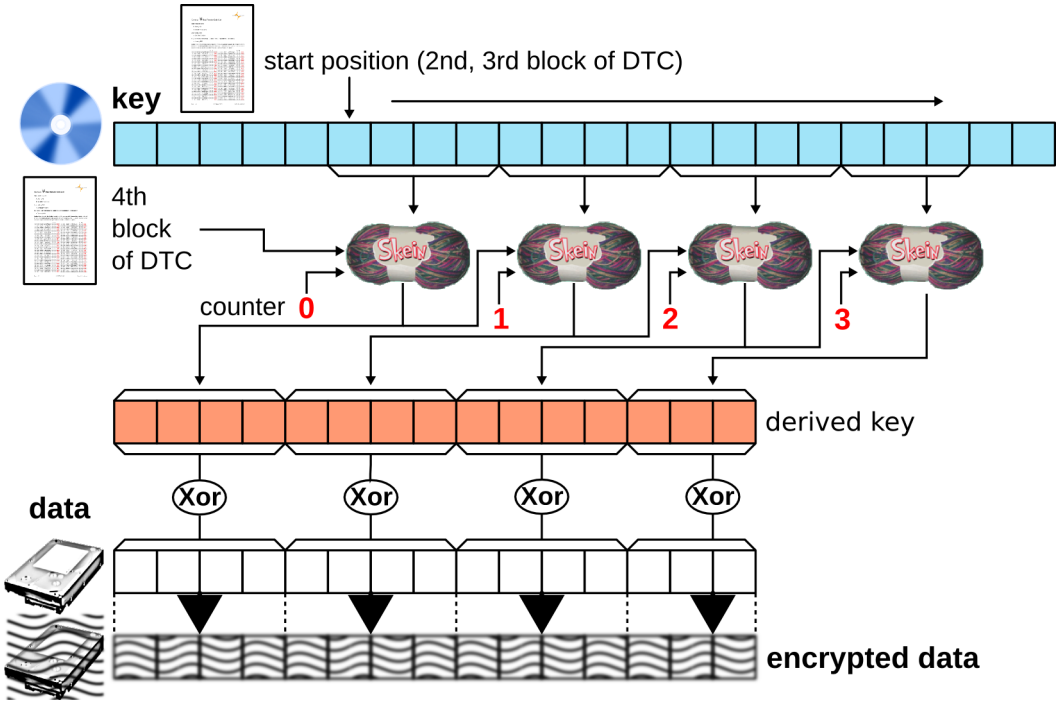

Figure 23: Encryption and decryption.

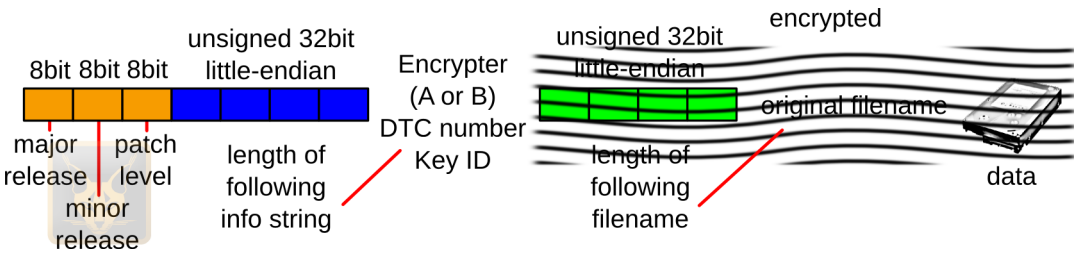

Figure 24: File format of encrypted data. The first byte to write/read is in the leftmost position.

sources as possible. Another pseudorandom number generator - even a bad one - incorporates additional noise. Permuting the pseudorandom sequence has the effect that the numbers do not appear sequentially. However, the best additional source of real entropy is the random string to be entered by the user. In the case an attacker influences the system provided random numbers this additional noise together with the permutation seems to yield good pseudorandom numbers. If used adequate a *jumping monkey* is hard to predict. The random string (4th block of DTC) for the encryption/decryption is an additional security facet that complicates any prediction. Before distribution, you may change these strings in the DTC list at will (but have to update the DTC check codes or you can do without that feature).

AZRAR is based on Skein [21] for many tasks. Skein is one of the best craptographic hash function family with provable security support [22]. The security of AZRAR is also based on the properties of Skein. Up to now there have been only two known cryptanalysis attack trials on Skein (as hash function). The first [33] tried to extend linear cryptanalysis to the block cipher in Skein, namely Threefish. This is not a security issue for Skein. The second [34] used a rotational rebound attack that only works if an attacker can manipulate both the plaintexts (unencrypted data) and the keys in a structured way, which is definitely uncritical to AZRAR. However, even if the attack only distinguishes reduced-round Threefish from a random permutation, it does not actually recover any bits. Threefish has a good security margin also in respect with this attack and, thus, the attack does not affect Skein. Nevertheless, the authors changed one constant in the algorithm's key schedule. Since version 1.3 of Skein this attack is ineffective.

Using the correct size of the data we could try to distinguish sensible short messages from pure random messages, if data follow certain building rules like source code or human language. The intended usage of AZRAR is the encryption and decryption of compressed archives like `zip`, `tar.gz`, etc.. Together with the encrypted original filename even the correct size of the data is hard to predict for an attacker. If not identifiable by the filename of the encrypted data even the compression algorithm is not known. However, even if an attacker predicts the correct compression approach all information about blocksizes etc. are encrypted preventing any prediction of the correct total size of the data. Hence, an attack which tries to separate sensible data from random data will be ineffective, if all data are compressed before encryption.

Version 1.2.1 of AZRAR uses 160 bit hash codes (with Skein-512) for integrity checking. Certainly, 1024 bit with Skein-1024 would be better with the negative effect of very long hash strings that are hard to compare manually. (Manual comparison is possible with intent). Main focus is to check integrity less forgery. Although not theoretically impossible, it is very unlikely that forged data exist, which can be decrypted with the intended key and DTC list, result in sensible data, and have the correct 160 bit hash code. An attacker that is able to succesfully construct such an unbelievable work of art must have access to all secret keys and the whole communication is broken. However, even if an attacker had access to all secret information it would be hard enough to construct sensible data that fulfill all criteria.

4 Logo and Name

AZRAR is an Egyptian name. It is the name of a caracal (*Caracal caracal*), also known as desert lynx. The caracal is a wild cat distributed across Africa, central Asia, and southwest Asia into India. Historically the caracal has been first categorized as Lynx due to the presence of a long tuft on the tip of the ear but it is more genetically related to the serval or the golden cat. The generic name caracal appeared first in 1843 after it has been misleadingly classified in 1776 as *Felis caracal*.

The caracal inhabits from semi-desert regions to thicket and moist woodland as well as evergreen and mountain forest areas. In the presence of trees some individuals use them to hide waiting for their prey. This fact led indirectly to the logo of this software – an amiable situation to its name.

5 License

Eveyone is permitted to copy and distribute copies of this manual as a whole, but changing it is not allowed. Distribution in extracts is not allowed without express written approval of Scientific Consilience.

AZRAR is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

AZRAR is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007
Copyright © 2007 Free Software Foundation, Inc. `http://fsf.org/`

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works. The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.
When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.
To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.
For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.
Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.
Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.
Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.
The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions

0. Definitions.

   "This License" refers to version 3 of the GNU General Public License.

   "Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

   "The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

   To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

   A "covered work" means either the unmodified Program or a work based on the Program.

   To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

   To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

   An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copy-

right notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

   The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

   A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

   The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

   The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

   The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

   The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

   All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

   You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide

you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

   No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

   When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   (a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

   (b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

   (c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

(d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

(a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

(b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

(c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

(d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

(e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

(a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

(b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

(c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

(d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

(e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

(f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

   You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

   Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

   You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

    Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

    An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

    You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge

for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity

of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

    If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

    Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

    The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

    If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

    THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

    IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

    If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>

Copyright (C) <textyear>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program>  Copyright (C) <year>  <name of author>

This program comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’.
This is free software, and you are welcome to redistribute it
under certain conditions; type ‘show c’ for details.
```

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see `http://www.gnu.org/licenses/`.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read `http://www.gnu.org/philosophy/why-not-lgpl.html`.

References

[1] G. Greenwald, E. MacAskill, L. Poitras, S. Ackerman, and D. Rushe. Microsoft handed the NSA access to encrypted messages. The Guardian online http://www.theguardian.com/world/2013/jul/11/microsoft-nsa-collaboration-user-data, July 2013.

[2] E. MacAskill. NSA paid millions to cover prism compliance costs for tech companies. The Guardian online http://www.theguardian.com/world/2013/aug/23/nsa-prism-costs-tech-companies-paid, August 2013.

[3] Reuters. Edward Snowden says NSA engages in industrial espionage. CBCNews online http://www.cbc.ca/news/world/edward-snowden-says-nsa-engages-in-industrial-espionage-1.2511635, January 2014.

[4] P. Zimmermann. *The official PGP User's Guide*. MIT Press, 1995.

[5] P. Zimmermann. *PGP Source Code and Internals*. MIT Press, 1995.

[6] J. Daemen and V. Rijmen. The design of Rijndael. AES: The Advanced Encryption Standard. In *Information Security and Cryptography*. Springer, 2002.

[7] A. J. Menezes, van Oorschot P. C., and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[8] B. Schneier. *Applied Cryptography*. John Wiley & Sons, 1996.

[9] A.K. Lenstra. Integer factoring. *Designs, Codes and Cryptography*, 19(2–3):101–128, March 2000.

[10] R. P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, 1982.

[11] D. R. Simon. On the power of quantum computation. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 116–123, 1994.

[12] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM: Journal on Computing*, 26:1484–1509, 1997.

[13] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang. Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance. *Nature*, 414(6866):883–887, 2001.

[14] F. Miller. *Telegraphic Code to Insure Privacy and Secrecy in the Transmission of Telegrams*. C.M. Cornwell, 1882.

[15] S.M. Bellovin. Frank Miller: Inventor of the one-time pad. *Cryptologia*, 3(35), 2011.

[16] C. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28:656–715, 1949.

[17] P. Beuth. Stuxnet war der erste Test der NSA-Wanze Cottonmouth. ZEIT-ONLINE http://www.zeit.de/digital/datenschutz/2014-01/nsa-wanzen-stuxnet, Januar 2014.

[18] D. E. Sanger and T. Shanker. N.S.A. devises radio pathway into computers. The New York Times online http://www.nytimes.com/2014/01/15/us/nsa-effort-pries-open-computers-not-connected-to-internet.html?ref=technology&_r=1&gwh=D5613EA72D2D79BB43A413748ECE7F76&gwt=pay, January 2014.

[19] R. Nixon. U.S. postal service logging all mail for law enforcement. The New York Times online http://www.nytimes.com/2013/07/04/us/monitoring-of-snail-mail.html?hp&_r=1&, July 2013.

[20] C. Savage. N.S.A. said to search content of messages to and from U.S. The New York Times online http://www.nytimes.com/2013/08/08/us/broader-sifting-of-data-abroad-is-seen-by-nsa.html?_r=0, August 2013.

[21] N. Ferguson, S. Lucks, B. Schneier, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker. The Skein hash function family. Online https://www.schneier.com/skein1.3.pdf, October 2010.

[22] M. Bellare, T. Kohno, S. Lucks, N. Ferguson, B. Schneier, D. Whiting, J. Callas, and J. Walker. Provable security support for the Skein hash family. Online https://www.schneier.com/skein-proofs.pdf, April 2009.

[23] B. Schneier. Description of a new variable-length key, 64-bit block cipher (Blowfish). In *Fast Software Encryption, Cambridge Security Workshop Proceedings (December 1993)*, pages 191–204. Springer, 1994.

[24] J. Kelsey, B. Schneier, D. Wagner, and C. Hall. Cryptanalytic attacks on pseudorandom number generators. In *Fast Software Encryption, Fifth International Workshop Proceedings (March 1998)*, pages 168–188. Springer, 1998.

[25] J. Kelsey, B. Schneier, C. Hall, and D. Wagner. Secure applications of low-entropy keys. In *1997 Information Security Workshop (ISW'97)*, pages 121–134, September 1997.

[26] M. Blaze, W. Diffie, R. Rivest, B. Schneier, T. Shimomura, E. Thompson, and M. Weiner. Minimal key lengths for symmetric ciphers to provide adequate commercial security. Online https://www.schneier.com/paper-keylength.html, January 1996.

[27] J. Kelsey, B. Schneier, and N. Ferguson. Yarrow-160: Notes on the design and analysis of the yarrow cryptographic pseudorandom number generator. In *Sixth Annual Workshop on Selected Areas in Cryptography*. Springer, August 1999.

[28] N. Ferguson, B. Schneier, and T. Kohno. *Cryptography Engineering*. John Wiley & Sons, 2010.

[29] J. A. Buchmann. *Introduction to Cryptography*. Springer, 2001.

[30] A. S. Tanenbaum and T. M. Austin. *Structured Computer Organization*. Prentice Hall PTR, August 2012.

[31] Y. Tsunoo, T. Saito, H. Kubo, M. Shigeri, T. Suzaki, and T. Kawabata. The most efficient distinguishing attack on VMPC and RC4A, 2005.

[32] A. Maximov. Two linear distinguishing attacks on VMPC and RC4A and weakness of RC4 family of stream ciphers (corrected). Cryptology ePrint Archive: Report 2007/070 http://eprint.iacr.org/2007/070, 2007.

[33] K. A. McKay and P. L. Vora. Pseudo-linear approximations for ARX ciphers with applications to Threefish. Online http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/Aug2010/documents/papers/MCKAY_PseudolinearApprox.pdf, August 2010.

[34] D. Khovratovich, I. Nikolié, and C. Rechberger. Rotational rebound attacks on reduced Skein. Online http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/Aug2010/documents/papers/RECHBERGER_rot-rebound.pdf, August 2010.